目　　录

第 1 章　MATLAB 语言入门

本章要点：

☑ MATLAB 语言发展历程
☑ MATLAB 7.0 语言主要特点
☑ MATLAB 7.0 语言平台介绍

1.1　MATLAB 语言的发展历程与主要特点

1.1.1　MATLAB 的发展历程

MATLAB，取自矩阵（Matrix）和实验室（Laboratory）两个英文单词的前三个字母，意即“矩阵实验室”。它是一种以矩阵作为基本数据单元的程序设计语言，提供了数据分析、算法实现与应用开发的交互式开发环境，经历了 20 多年的发展历程。

20 世纪 70 年代中期，美国新墨西哥大学计算机系系主任 Clever Moler 博士和其同事在美国国家自然科学基金的资助下，开发了调用 LINPACK 和 EISPACK 的 Fortran 子程序，20 世纪 70 年代后期，Moler 博士编写了相应的接口程序，并将其命名为 MATLAB。

1983 年，John Little 和 Moler、Bangert 等一起合作开发了第 2 代专业版 MATLAB。1984 年，Moler 博士和一批数学专家、软件专家成立了 MATH WORKS 公司，继续 MATLAB 软件的研制与开发，并着力将软件推向市场。

1993 年，MATH WORKS 公司连续推出了 MATLAB 3.x（第 1 个 Windows 版本）、MATLAB 4.0。1997 年，MATH WORKS 公司推出了 MATLAB 5.0。

2001 年，MATH WORKS 公司推出了 MATLAB 6.x。2004 年，MATH WORKS 公司推出了 MATLAB 7.0。MATLAB 5.3 对应于 Release12，MATLAB 6.0 对应于 Release13，而 MATLAB 7.0 对应于 Release14。

MATLAB 分为总包和若干个工具箱，随着版本的不断升级，它具有越来越强大的数值计算能力、更为卓越的数据可视化能力及良好的符号计算功能，逐步发展成为各种学科、多种工作平台下功能强大的大型软件，获得了广大科技工作者的普遍认可。一方面，MATLAB 可以方便实现数值分析、优化分析、数据处理、自动控制、信号处理等领域的数学计算，另一方面，也可以快捷实现计算可视化、图形绘制、场景创建和渲染、图像处理、虚拟现实和地图制作等分析处理工作。在欧美许多高校，MATLAB 已经成为线性代数、自动控制理论、概率论及数理统计、数字信号处理、时间序列分析、动态系统仿真等课程的基本教学工具，是攻读学位本科生、研究生必须掌握的基本技能。在国内，这一语言也正逐步成为一些大学理工科专业学生的重要选修课。

1.1.2 MATLAB 的基本特点

MATLAB 是一种高级编程语言，其特点可以归纳为以下几点。

1．语言简单易学

MATLAB 是一种解释执行的语言，语句采取通用数学形式，语法规则与一般结构化高级编程语言（如 C 语言等）相差不大，并把编辑、编译、连接、执行功能融为一体，调试程序手段丰富、调试速度快，可以快速排除输入程序时书写、语法等方面的错误，具有一般语言基础的用户可以较快掌握。

2．代码短小高效

MATLAB 语言将矩阵作为最基本的数据单元、无须预先定义维数，函数是 MATLAB 中最基本、也是最重要的组成成分，而 MATLAB 将数学问题的许多算法编成了大量库函数、具有解决许多问题的工具箱，只要熟悉算法基本特点、函数调用格式和参数具体意义等内容，调用现成函数就可以较快解决自己专业领域的许多问题，而不必再花很多时间去实现常规算法，使得所编写的代码文件简单短小、求解专业问题时高效方便。

3．计算功能强大

MATLAB 语言具有强大的矩阵数值计算功能、可以方便地处理许多特殊矩阵，利用符号和函数可以对矩阵进行线性代数运算（加减乘除、转置和求逆等），适用于大型数值算法的编程实现；工具箱中许多高性能的数值计算算法，可以解决实际应用中的许多数学问题，尤其是与矩阵计算有关的问题。

4．绘图非常方便

MATLAB 语言具有强大的绘图功能，具有很多绘图函数命令，可以绘制一般的二维或三维图形（如线形图、条形图、饼图、散点图、直方图等），可以绘制工程特性较强的特殊图形（如玫瑰花图、极坐标图等），通过其可视化功能可以绘制一些用于数据分析的图形（如矢量图、等值线图、曲面图、切片图等），还可以生成快照并进行动画制作，使用 MATLAB 句柄图形对象并结合绘图函数可以绘制自己最为满意的图形，使用时只需调用不同的绘图函数、使得作图简单易行。

5．扩充能力强大

可扩展性是 MATLAB 的一个重要特点，MATLAB 通常包含系统本身定义的大量库函数，用户也可以定义自己的函数、组成自己的工具箱，不仅进行数学运算时可以直接调用、而且库函数名称与用户文件保持形式一致，用户可以根据需要方便地建立或扩充库函数、方便地解决本领域内的计算问题。MATLAB 提供了与 Fortran、C\C++语言及一些应用程序（如 Excel）的接口，利用 MATLAB 编译器和运行服务器还可以生成独立的可执行程序，使用户可以混合编程、也可以隐藏算法并避免依赖 MATLAB 平台环境。

6．帮助功能完整

MATLAB 采用基于 HTML 的自述文件，自述文件中不仅介绍了 MATLAB 语言，还对各种算法的理论基础与算法实现进行了比较详细的说明、并给出了相应的常规实例，帮助功能比较完整，用户使用较为方便。

1.1.3 MATLAB 7.0 的新特点

MATLAB 7.0 可在下列平台上安装：

- Windows 2000（SP3 或 SP4）
- Windows NT 4.0（SP5 或 SP6a）
- Windows XP
- Linux ix86 2.4.x, glibc 2.2.5
- Sun Solaris 2.8 and 2.9
- HPUX 11.0 and 11.i
- Mac OS X 10.3.2

MATLAB 7.0 的新特点主要包括以下几个方面。

1．开发环境

- 对桌面进行了重新设计，提供了多文档管理、锚定图形窗口及保存定制输出和常用命令快捷键的命令；
- 改进了数组编辑器和工作区间浏览器，使得查看、编辑变量和使用变量数据绘制图形更加容易；
- 可将程序代码发布为 HTML、C/C++、Java、Word 等格式的文档；
- 命令窗口中有关帮助命令可以与自述文件的对应部分直接链接，同时自述文件中增加了一些动态演示。

2．编程

- 可以创建嵌套函数，提供定义和调用自定义函数更为便捷的途径；
- 提供在命令行和命令文件中定义单行函数的隐函数表示形式；
- 使用条件断点，可以在条件表达式为真时停止运行。

3．计算

- 整数计算部分可以使用户处理更大的整型数据集；
- 单精度计算、FFT、线性代数和滤波设计部分使用户可以处理更大的单精度数据集；
- 计算几何部分对算法选择给出了更多控制；
- 使用 linsolve 函数，通过指定矩阵系数结构，可以更快求解线性方程组；
- ODE 求解器可以控制隐式差分方程和多点边值问题。

4．图形和三维可视化实现

- 使用新的绘图界面，可以在不输入程序代码的情况下交互式地创建和编辑图形；
- 可以生成图形的程序代码，利用该代码可以重建图形；
- 一些特殊图形更易修改；
- 改进了图形标注，包括绘制图形、对象对齐和将标注"钉"到数据点；
- 提供了数据勘查工具，可以更方便地进行图形平移和数据光标等；
- 可以对成组图形对象进行变换；
- 可以在 GUIDE 中对用户界面面板和控件进行成组控制。

5．数据获取和外部接口

- 提供读取很大文本文件和写为 Excel 等文件格式的命令函数；
- 提供压缩 MAT 文件的选项，可以用更少的磁盘空间保存较大的数据；
- 使用 javaaddpath 函数可以在不重新启动 MATLAB 的情况下动态添加、删除和重载 Java 类；
- COM 定制接口、服务器事件和 Visual Basic 脚本支持。

✦ 可以基于 SOAP（Simple Object Access Protocol，简单对象存取协议）获取 Web 服务；
✦ 提供可以连接到 FTP 服务器进行远程文件操作的 FTP 对象。

此外，与 MATLAB 6.0 相比，MATLAB 7.0 对下述工具箱进行了改进或赋予了新内容：

✦ 通信系统工具箱（Communications Toolbox）
✦ 控制系统工具箱（Control System Toolbox）
✦ 数据库工具箱（Database Toolbox）
✦ 滤波设计工具箱（Filter Design Toolbox）
✦ 金融分析工具箱（Financial Derivatives Toolbox）
✦ 仪表控制工具箱（Instrument Control Toolbox）
✦ 图像处理工具箱（Mapping Toolbox）
✦ 模型预测控制工具箱（Model Predictive Control Toolbox）
✦ 优化工具箱（Optimization Toolbox）
✦ 统计工具箱（Statistics Toolbox System）
✦ 系统辨识工具箱（Identification Toolbox）
✦ 虚拟现实工具箱（Virtual Reality Toolbox）
✦ 小波工具箱（Wavelet Toolbox）

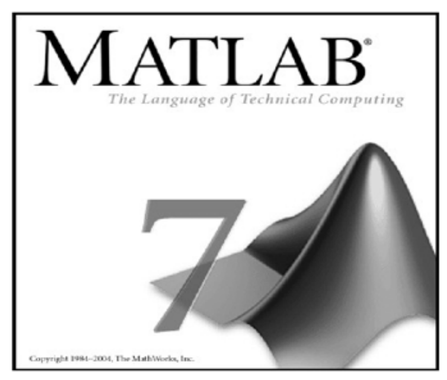
1.2 MATLAB 7.0 语言平台介绍

1.2.1 MATLAB 7.0 语言平台安装

（1）插入 MATLAB 7.0 光盘即可自动启动安装，启动界面如图 1-1 所示。



（a）准备安装文件



（b）启动界面

图 1-1 MATLAB 7.0 安装启动界面

（2）启动安装后，出现欢迎窗口，如图 1-2 所示。选择【Install】后单击【Next】按钮。

（3）输入用户姓名、单位、系列号，如图 1-3 所示，单击【Next】按钮。

（4）出现注册许可信息窗口，如图 1-4 所示，可以选择【Yes】，并单击【Next】按钮。

（5）出现定制与典型安装选择窗口，如图 1-5 所示，这里可以选择【Typical】（典型安装），并单击【Next】按钮。

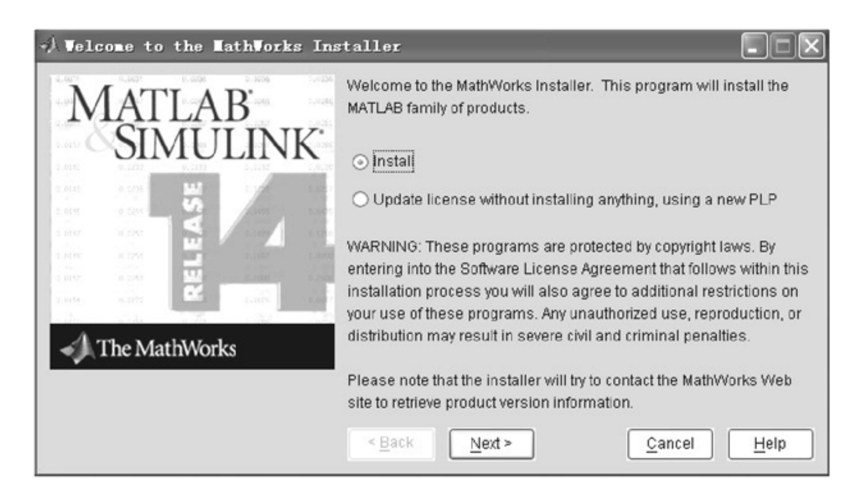


图 1-2　欢迎安装对话框

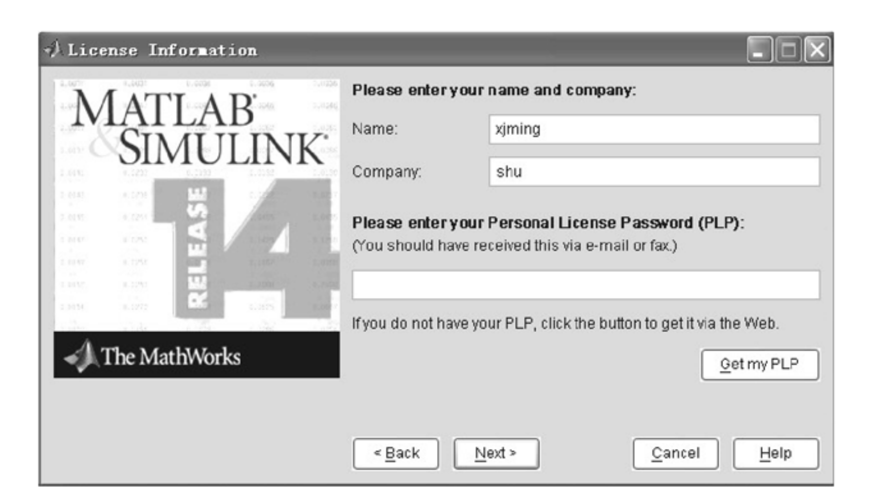


图 1-3　输入用户姓名、单位与系列号

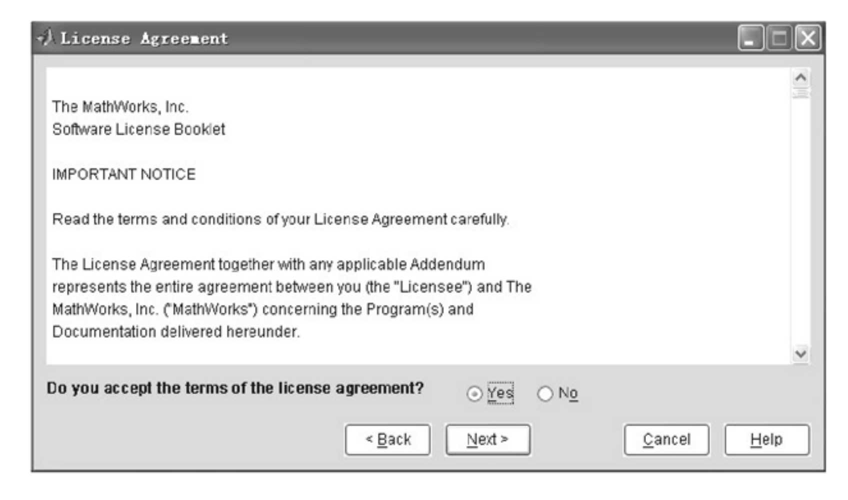


图 1-4　注册许可信息窗口

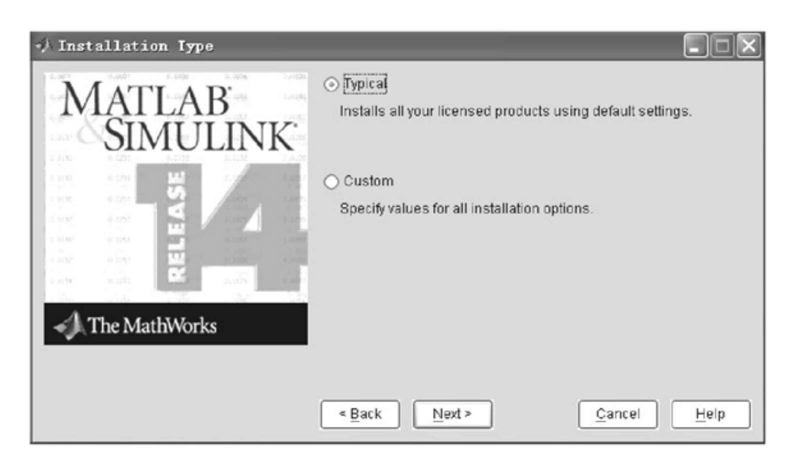
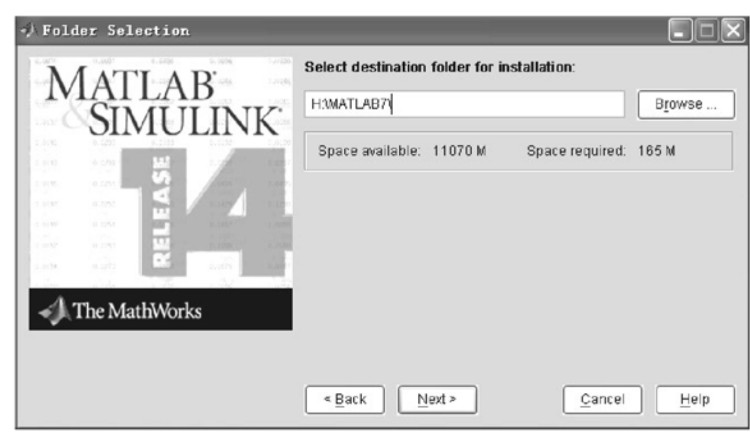
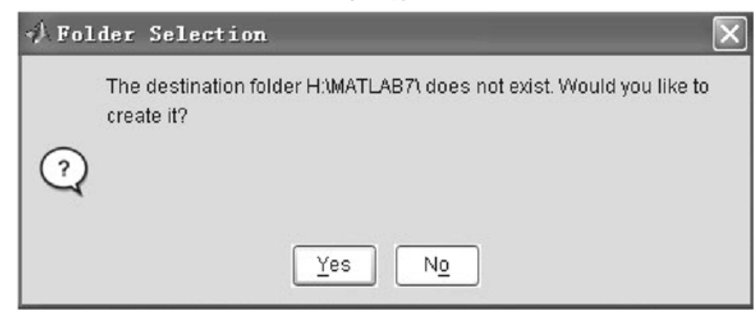


图 1-5　定制与典型安装选择窗口

（6）出现安装目录窗口，如图 1-6（a）所示，可以选择"C:\MATLAB7\"（默认）、也可以选择其他目录，并单击【Next】。若安装位置目录不存在可创建新目录，之后单击【Yes】，如图 1-6（b）所示。



（a）安装目录窗口



（b）创建安装目录对话框

图 1-6　确定安装目录

（7）出现确认安装信息窗口，如图 1-7 所示，浏览确认后单击【Install】按钮。

（8）显示安装进度，如图 1-8 所示。

（9）安装过程中出现如图 1-9 所示的提示，放入第 2 张光盘后单击【OK】，否则选【Skip CD 2】。

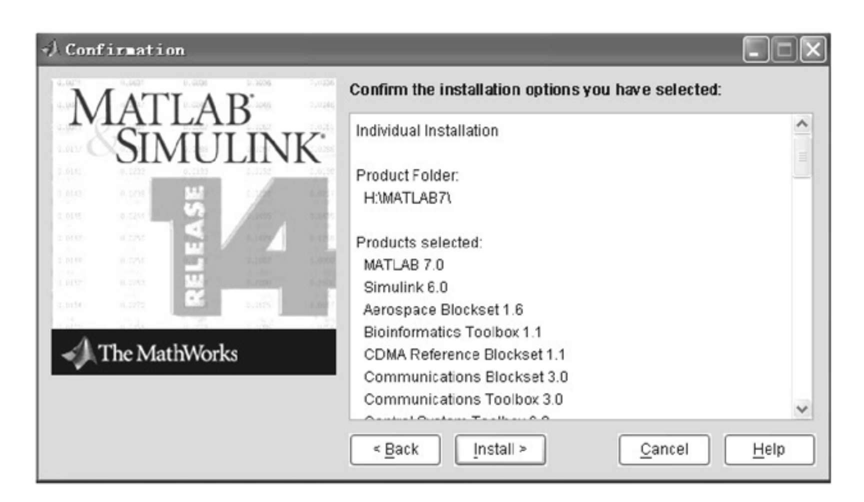


图 1-7　确定安装信息窗口

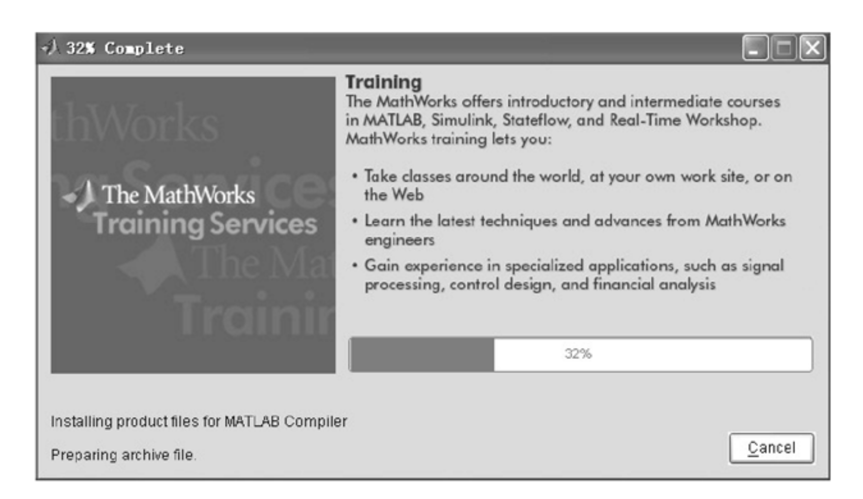


图 1-8　显示安装进度



图 1-9　安装 MATLAB 7.0 信息提示

（10）出现设置安装信息提示对话框，如图 1-10 所示，单击【Next】按钮。

（11）安装结束并提示是否立即运行 MATLAB 7.0（默认为立即运行），如图 1-11 所示，可以不选【Start MATLAB】而是单击【Finish】按钮。

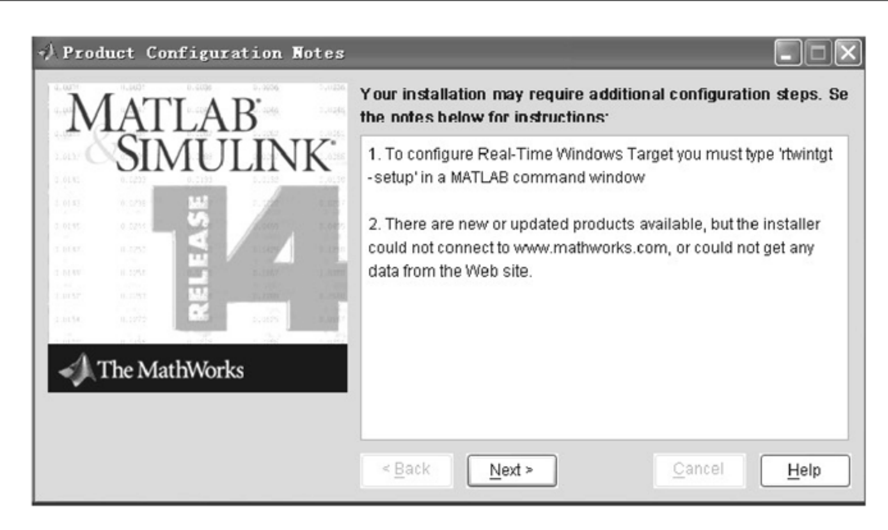


图 1-10　安装 MATLAB 7.0 Web 服务器信息

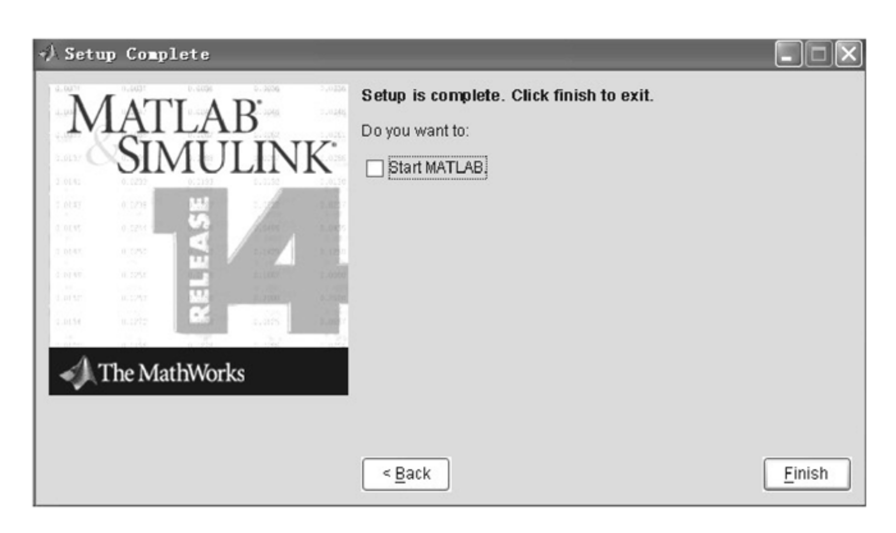


图 1-11　安装结束

1.2.2　MATLAB 7.0 语言平台介绍

1．程序编制步骤

（1）进入编译平台。在 Windows XP 平台上（其他平台与此类似），双击桌面图标或者选择【开始】→【程序】→【MATLAB 7.0】→【MATLAB 7.0】，均可进入 MATLAB 7.0 编译平台，如图 1-12 所示。

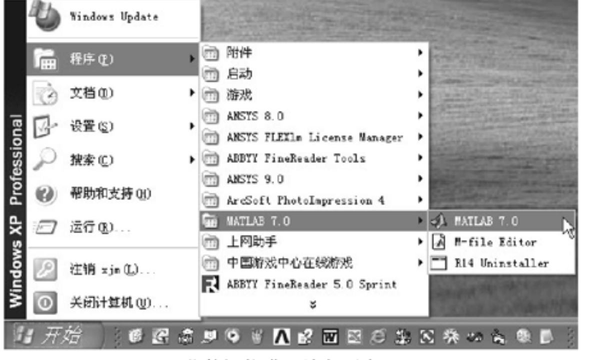
（2）编译 m 文件或通过命令窗口输入适当的函数命令。

（3）若使用图形用户界面编程，则设计 MATLAB 7.0 下可视化程序界面（加入控件、对有关属性进行设置等）并编制相应的 m 文件。

（4）m 文件有命令文件和函数文件两种形式，命令文件的变量均为全局变量且无参数传递，而函数文件一般由 function 命令开始，变量通常是局部变量，可传递多个输入输出参数。



（a）从桌面快捷方式进入编程平台



（b）菜单操作进入编程平台

图 1-12　进入 MATLAB 7.0 编程平台

（5）执行编译过程并修改完善程序。

2．编程界面介绍

MATLAB 7.0 编程界面各部分的名称如图 1-13 所示。



图 1-13　MATLAB 7.0 编程界面各部分组成名称

3．使用帮助文件

MATLAB 7.0 的函数命令很多，全部记住既不需要也不可能。这些函数可通过以下两种方法学习。

（1）使用函数在线帮助。通常使用命令 help，比如要知道函数命令 sin 的含义、格式、实例，可在命令窗口键入 help sin。对于已知函数命令名称、但不熟练其具体使用方法的命令来说，使用 help 命令时，函数命令名称通常都是小写字母。

（2）使用全部帮助。在 MATLAB 7.0 的自述文件中给出了 MATLAB 7.0 中的全部帮助，包括 MATLAB 7.0 语言介绍、函数命令含义与算法、工具箱说明、典型算例等，如图 1-14 所示。

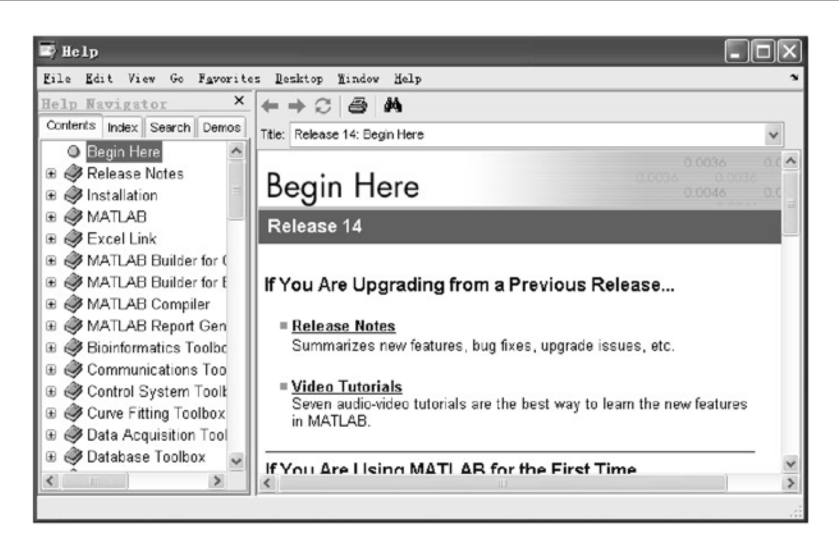


图 1-14 MATLAB 7.0 语言自述帮助文件

习题 1

1．简述 MATLAB 7.0 的特点。
2．简述 MATLAB 7.0 的安装过程。
3．简述 MATLAB 7.0 编程平台各部分的名称。
4．简述 MATLAB 7.0 的程序编制步骤。
5．简述 MATLAB 7.0 中帮助函数的使用方法。

第 2 章　MATLAB 7.0 程序设计基础

本章要点：

- ☑ MATLAB 中特殊常量与一般变量的命令与使用方法；
- ☑ MATLAB 中各种数组的建立与引用方法；
- ☑ MATLAB 中各种运算符（算术运算符、关系运算符、逻辑运算符）的使用方法；
- ☑ MATLAB 中 m 文件与 m 函数的建立与调用方法；
- ☑ MATLAB 的程序结构（包括顺序结构、分支结构与循环结构）。

2.1　常量与变量

2.1.1　特殊常量

MATLAB 语言本身具有一些固定变量，这些特殊的固定变量称为常量。这些常量具有特定的意义，用户在自定义变量名时应避免使用。表 2-1 给出了 MATLAB 语言中常用的一些常量。

表 2-1　MATLAB 语言中的常量

常　量　名	常　量　值
ans	用于结果的默认变量名
pi	圆周率
inf	无穷大，如 2/0
NaN	Not-a-number，表示不定值
tic	秒表开始执行
toc	秒表停止
i,j	虚数单位
date	日历
clock	挂钟

2.1.2　一般变量（可动变量）

变量是任何程序设计语言的基本单位，MATLAB 语言当然也不例外。与一般程序设计语言不同的是，MATLAB 语言并不要求事先对所使用变量进行声明，也不需要指定变量类型，MATLAB 语言自动依据变量值或对变量操作来识别变量类型。在赋值过程中，如果赋值变量已存在时，MATLAB 语言将使用新值代替旧值，并以新值类型代替旧值类型。

在 MATLAB 语言中变量的命名应遵循如下规则。

（1）变量名区分大小写。

（2）变量名长度不超过 31 位，第 31 个字符之后的字符将被 MATLAB 语言所忽略。

（3）变量名以字母开头，可以由字母、数字、下划线组成，但不能使用标点。

MATLAB 语言中也存在变量作用域的问题。在未加特殊说明的情况下，MATLAB 语言将所识别的一切变量视为局部变量，即仅在其使用的 m 文件内有效。要将变量定义为全局变量，应当对变量进行说明（在该变量前加关键字 global）。一般来说，全局变量字符用大写英文字母表示。

2.2 数组

数组是 MATLAB 语言中较为简单的数据组织形式，在数值计算中的应用十分广泛，它可以视为矩阵的一种特殊表现形式，也可以通过矩阵理解数组操作。

2.2.1 建立数组

MATLAB 中建立一维和二维数组，常用逐个元素输入法、冒号法或一些特殊方法，下面分别讲述。

1．逐个元素输入法

[例 2-1] 生成数组 x，其中：

$$x = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 3 & 4 \\ 2 & 4 & 5 \end{bmatrix}$$

[算例代码]

```
%例 2-1
x=[1 2 3;2 3 4;2 4 5]
```

[运行结果]

```
x =
    1    2    3
    2    3    4
    2    4    5
```

2．冒号法

[调用格式]

x= 初始量 : 步长 : 终止量

[例 2-2] 生成数组 x，其中 x = [1, 2, 3, 4, 5]。

[算例代码]

```
%例 2-2
x=1:1:5
```

[运行结果]

```
x =

     1     2     3     4     5
```

3．特殊方法

[调用格式]

x=linspace（初始量, 终止量,数组元素个数）

y=logspace（初始量, 终止量,数组元素个数）

重要提示：一般来说，logspace（y1,y2,N）表示在 10^y1 和 10^y2 之间插入 *N*–2 个元素，组成一个含有 *N* 个元素的数组；如果 *y2*=pi，则表示在 10^y1 和 pi 之间插入 *N*–2 个元素；如果 *N*<2，返回值为 10^y2。

[例 2-3]　生成数组 x，其中 x = [1, 2, 3, 4, 5]。

[算例代码]

```
%例 2-3
x=linspace(1, 5,5)
```

[运行结果]

```
x =

     1     2     3     4     5
```

[例 2-4]　生成数组 y=logspace(0,2,5)。

[算例代码]

```
%例 2-4
y=logspace(0,2,5)
```

[运行结果]

```
y =

    1.0000    3.1623   10.0000   31.6228  100.0000
```

[例 2-5]　生成数组 y=logspace(1, pi,5)。

[算例代码]

```
%例 2-5
y=logspace(1, pi,5)
```

[运行结果]

```
y =

   10.0000    7.4866    5.6050    4.1963    3.1416
```

2.2.2　引用数组

1．一维数组

[调用格式]

x(n)；x(n1:n2)

x(n)表示一维数组中的第 *n* 个元素；x(n1:n2)表示一维数组中的第 *n*1 至 *n*2 个元素。

[例 2-6] 已知一维数组 x = [1, 2, 3, 4, 5]，求 x(1:3)。

[算例代码]

```
%例 2-6
x = [1, 2, 3, 4, 5];
x(1:3)
```

[运行结果]

```
ans =
     1     2     3
```

2．二维数组

[调用格式]

x(m,:)；x(:,n)；x(m,n1:n2)

x(m,:)表示二维数组中的第 *m* 行元素；x(:,n)表示二维数组中的第 *n* 列元素；x(m,n1:n2)表示二维数组中第 *m* 行中的 *n*1 至 *n*2 个元素。

[例 2-7] 二维数组 x=[1 2 3;2 3 4;2 4 5]，求 x(2,2:3)。

[算例代码]

```
%例 2-7
x=[1 2 3;2 3 4;2 4 5];
x(2,2:3)
```

[运行结果]

```
ans =
     3     4
```

2.2.3 细胞数组与结构数组

1．细胞数组

细胞数组是以单元为元素的数组，每个元素称为单元，每个单元可以包含其他类型数组，如实数矩阵、字符串数组、复数向量。细胞数组通常由{}创建，其数据通过数组下标引用。

[例 2-8] 创建一个 2×2 的细胞数组 A。

[算例代码]

```
%例 2-8
clear
A(1,1)={3+2i};
A(1,2)={'time'};
A(2,1)={[1 2 3 4]};
A(2,2)={[1 2;3 4]};
A
```

[运行结果]

```
A =
    [3.0000+ 2.0000i]      'time'
        [1x4 double]      [2x2 double]
```

2．结构数组

结构数组是根据属性名（field）组织起来的不同类型数据的集合。结构的任何一个属性可以包含不同数据类型，如字符串文本、标量、矩阵等。MATLAB 使用分级存储机制来存储不同类型的数据，结构数组数据通过属性名来引用。

[函数命令]

```
struct
```

[调用格式]

```
s = struct('field1', {}, 'field2', {}, …)
s = struct('field1', values1, 'field2', values2, …)
```

s = struct('field1', {}, 'field2', {}, …)表示建立一个空的结构数组，不含数据。

s = struct('field1', values1, 'field2', values2, …)表示建立一个具有属性名和数据的结构数组。

重要提示：当扩展结构数组时，MATLAB 对未指定的属性以空矩阵赋值，且数组中每个结构具有同样多的属性名、所有属性具有相同的属性名。

[例 2-9]　结构数组 student=struct('name',{'Liu','Wang'},'Age',{'20','21'})，求 student(1), student(2)，student(2).name。

[算例代码]

```
%例 2-9
student=struct('name',{'Liu','Wang'},'Age',{'20','21'})        %建立一个维数是[1 2]的数组
student(1)                                                      %查看数组第 1 个结构的数据
student(2)                                                      %查看数组第 2 个结构的数据
student(2).name                                                 %通过属性名引用数据
```

[运行结果]

```
student =
        1x2 struct array with fields:
            name
            Age
ans =
    name: 'Liu'
    Age: '20'
ans =
    name: 'Wang'
    Age: '21'
ans =
    Wang
```

2.3 运算符

2.3.1 算术运算符

MATLAB 语言的算术运算符如表 2-2 所示。

表 2-2 MATLAB 语言中的算术运算符

操作符	意 义
+	算术加
–	算术减
*	算术乘
.*	点乘
^	算术乘方
.^	点乘方
\	算术左除
.\	点左除
/	算术右除
./	点右除

其中，算术加、减、乘及乘方与传统意义的加、减、乘及乘方相类似，用法也基本相同。而点乘、点乘方等运算则有其特殊的一面，点运算是指元素点对点运算，即矩阵内元素对元素之间的运算，点运算要求参与运算的变量在结构上必须是相似的。

相对而言，MATLAB 语言中除法运算较为复杂。对于简单数值运算而言，算术左除与算术右除也不同，算术右除与传统除法相同，即 $a/b=a\div b$；而算术左除则与传统除法相反，即 $a\backslash b=b\div a$。对于矩阵而言，算术右除 $\boldsymbol{B}/\boldsymbol{A}$ 相当于求解线性方程组 $x\boldsymbol{B}=\boldsymbol{A}$，即 $\boldsymbol{B}/\boldsymbol{A}=\boldsymbol{A}\times \mathrm{inv}(\boldsymbol{B})$；而算术左除 $\boldsymbol{B}\backslash\boldsymbol{A}$ 则相当于求解线性方程组 $\boldsymbol{A}x=\boldsymbol{B}$，即 $\boldsymbol{B}\backslash\boldsymbol{A}=\mathrm{inv}\ \boldsymbol{A}\times(\boldsymbol{B})$；点左除与点右除与前面所述点运算类似，是变量对应元素进行数值点除。

[例 2-10] 矩阵 $\boldsymbol{A}$=[1 2 ;2 3]，$\boldsymbol{B}$=[1 0;2 2]，求 A.*B。

[算例代码]

```
%例 2-10
A=[1 2 ;2 3];
B=[1 0;2 2];
A.*B
```

[运行结果]

```
ans =
      1     0
      4     6
```

2.3.2 关系运算符

MATLAB 语言的关系运算符如表 2-3 所示。

表 2-3　MATLAB 语言中的关系运算符

操　作　符	定　　义
==	等于
~=	不等于
>	大于
>=	大于等于
<	小于
<=	小于等于

关系运算符主要用来对矩阵与数、矩阵与矩阵进行比较，返回二者关系的、由数 0 和 1 组成的矩阵，0 和 1 分别表示不满足和满足指定关系。判断一个矩阵是否为空矩阵时，一般不能使用“==”，而应当使用函数 isempty，MATLAB 的空矩阵是指矩阵存在，但不含任何元素。

[例 2-11]　矩阵 ***A***=[1 2 ;2 3]，求 ***A*** 中等于 2 的元素个数 *n*。

[算例代码]

```
%例 2-11
A=[1 2 ;2 3];
B=A==2
n=sum(sum(B))
```

[运行结果]

```
B =
     0     1
     1     0
n =
     2
```

2.3.3　逻辑运算符

MATLAB 语言的逻辑运算符如表 2-4 所示。

表 2-4　MATLAB 语言中的逻辑运算符

操　作　符	定　　义
&	逻辑与
|	逻辑或
~	逻辑非
xor	逻辑异或

MATLAB 语言中进行逻辑判断时，所有非零数值均被认为为真，而零为假；在逻辑判断结果中，判断为真时输出 1、判断为假时输出 0。

在算术、关系、逻辑三种运算符中，算术运算符优先级最高，关系运算符次之，而逻辑运算符的优先级最低。在逻辑“与”、“或”、“非”三者中，“与”和“或”有相同的优先级，从左到右依次执行都低于“非”的优先级。实际应用中可以通过括号来调整运算过程中的次

序。

对两个标量 *a* 和 *b* 进行逻辑运算时，运算规则如表 2-5 所示。

表 2-5 MATLAB 语言中的逻辑运算规则

输 入		与	或	异或	非
a	b	a&b	a|b	xor(a,b)	～a
0	0	0	0	0	1
0	1	0	1	1	1
1	0	0	1	1	0
1	1	1	1	0	0

[例 2-12] 逻辑矩阵 ***A***=[1 1; 0 1]、***B***=[0 1; 0 0]，逻辑标量 *b*=0，求 C1=A&b, C2=A|b, C3=xor(A,B)。

[算例代码]

```
%例 2-12
A=[1 1 ;0 1];
b=0;
B=[0 1;0 0];
C1=A&b
C2=A|b
C3=xor(A,B)
```

[运行结果]

```
C1 =
        0    0
        0    0
C2 =
        1    1
        0    1
C3 =
        1    0
        0    1
```

2.3.4 逻辑函数

逻辑函数非常有用，在交互运算、矩阵变化时可以方便地查找或替换矩阵中满足一定条件的部分或所有元素，使用过程中只有认真体会每个函数的具体用法、才能在实际应用中灵活运用。MATLAB 中的主要逻辑函数如表 2-6 所示。

表 2-6 MATLAB 语言中的逻辑函数

逻 辑 函 数	函 数 意 义
all	判断是否所有元素为非零值
any	判断是否存在一个元素为非零值

续表

逻 辑 函 数	函 数 意 义
exist	查看变量或函数是否存在
find	找出向量或矩阵中非零元素的位置标识
isempty	判断矩阵是否为空矩阵
isequal	判断几个对象是否相等
isnumeric	判断对象是否为数值型

对于二维矩阵 ***A***，all(A)表示如果矩阵 ***A*** 的某列所有元素都为非零数，则返回结果的当前列为逻辑“真”，即逻辑值 1。逻辑函数 any 的用法与 all 相同，其在矩阵相除时，可判断除数矩阵是否有零元素。使用 isnumeric 等命令时，如果判断对象为系数矩阵、双精度矩阵、复数矩阵等，则返回值为 1，如果判断对象为字符串、结构矩阵等，则返回值为 0。在程序设计中，有时需要知道变量是否已被定义过，有时需要了解变量的类型，这时 exist 函数非常有用，其调用格式为 a =exist('A')，a 的返回值为数 0～7、分别表示不同的含义，如表 2-7 所示。

表 2-7　函数 exist 返回值和变量状态或类型的对应关系

a	***A*** 的状态或类型
0	对象 *A* 不存在或不在 MATLAB 的搜索路径下
1	对象 *A* 是工作空间中的一个变量
2	对象 *A* 是一个 M 文件或在 MATLAB 搜索路径下未知类型的文件
3	对象 *A* 是一个 MATLAB 搜索路径下的 MEX 文件
4	对象 *A* 是一个 MATLAB 搜索路径下的已编译的 SIMULINK 函数（MDL 文件）
5	对象 *A* 是 MATLAB 的内置函数
6	对象 *A* 是一个 MATLAB 搜索路径下的 P 文件
7	对象 *A* 是一个路径，不一定是 MATLAB 搜索路径

[例 2-13]　向量 ***a***=[1 3 5]，矩阵 ***A***=[1 2 3 ;0 4 5]，求 B1=all(a)，B2=all(A)，B3=all(A,2)。

[算例代码]

```
%例 2-13
a=[1 3 5];
A=[1 2 3 ;0 4 5];
B1=all(a)                                %判断向量 a 中的每个元素是否为非零数
B2=all(A)                                %将矩阵 A 按列进行操作
B3=all(A,2)                              %将指定的第 2 维作为向量进行运算
```

[运行结果]

```
B1 =
     1
B2 =
     0     1     1
B3 =
     1
     0
```

[例 2-14] 向量 *a*=[0 3 0]，矩阵 ***A***=[1 0 3 ;0 0 5]，求 B1=any(a)，B2=any(A)，B3=any(A,2)。

[算例代码]

```
%例 2-14
a=[0 3 0];
A=[1 0 3 ;0 0 5];
B1=any(a)
B2=any(A)
B3=any(A,2)
```

[运行结果]

```
B1 =

     1

B2 =

     1     0     1

B3 =

     1
     1
```

[例 2-15] 求返回值 a1=exist('work')，a2=exist('filtdes')，a3=exist('c:\windows')。

[算例代码]

```
%例 2-15
a1=exist('work')              % work 是 MATLAB 7.0 目录下的一个子目录
a2=exist('filtdes')           % filtdes 是 toolbox\signal 目录下的一个文件
a3=exist('c:\windows')        % windows 是不在 MATLAB 7.0 目录下的一个子目录
```

[运行结果]

```
a1 =

     7

a2 =

     6

a3 =

     7
```

[例 2-16] 矩阵 ***A***=[1 0 3 ;0 0 5]，求 k=find(A)，[i,j]=find(A)，[i,j,v]=find(A)。

[算例代码]

```
%例 2-16
A=[1.2 0 3.5 ;0 0 5.4];
k=find(A)                     %标识按列进行，即从第 1 列开始依次数下去
[i,j]=find(A)                 % i 表示行标，j 表示列标
[i,j,v]=find(A)               % v 为相应的非零元素的值
```

[运行结果]

```
k =

    1
```

```
     5
     6
i =                  j =
     1                    1
     1                    3
     2                    3
i =                  j =                  v =
     1                    1                    1.2000
     1                    3                    3.5000
     2                    3                    5.4000
```

[算例说明]　对于大型矩阵，需要查找符合一定条件的元素时，用 find 函数更为方便。

[例 2-17]　矩阵 ***A***=[0.34 0.6]，***B***=[0.34 ;0.6]，***C***=['who']，判断 ***A*** 与 ***B***、***A*** 与 ***C*** 是否相等。

[算例代码]

```
%例 2-17
A=[0.34 0.6];
B=[0.34 ;0.6];
C=['who'];
isequal(A,B)
isequal(A,C)
```

[运行结果]

```
ans =
     0
ans =
     0
```

重要提示：当判断对象具有相同的类型、维数和内容时，返回 1，否则返回 0。

[例 2-18]　矩阵 ***A***=[0.34 0.6]，***B***=[1+2**i* 0.6+3*i]，***C***=['who']，判断 ***A***、***B***、***C*** 是否为数据矩阵。

[算例代码]

```
%例 2-18
A=[0.34 0.6];                 %  A 为实数矩阵
B=[1+2*i 0.6+3*i];            %  B 为复数矩阵
C=['who'];                    %  C 为字符串
isnumeric(A)
isnumeric(B)
isnumeric(C)
```

[运行结果]

```
ans =
     1
ans =
     1
```

```
ans =
    0
```

2.3.5 字符串操作

1．字符串定义

[调用格式]

s='str'，name=['str1' 'str2' 'str3']

[例 2-19] 定义字符串 s1=['who' 'are' 'you']，s2=['I''m' 'Ding' 'Tao']，s3=[s1;s2]

[算例代码]

```
%例 2-19
s1=['who'   'are'   'you']
s2=['I''m'   'Ding'   'Tao']
s3=[s1 s2]
m=size(s1)
n= size(s2)
```

[运行结果]

```
s1 =
    whoareyou
s2 =
    I'mDingTao
s3 =
    whoareyou
    I'mDingTao
m =
    1      9
n =
    1      10
```

重要提示：与数组不同，字符串矩阵中每一行字符串元素的个数可以不同、但字符总个数必须相同；单个字符串中字符之间的空格也算一个字符；在字符串中要输入“'”必须通过两个“'”实现。

2．字符串转换

（1）字符串转换方式一。

[函数命令]

char

[调用格式]

S=char(T)

S=char(T)表示将由正整数组成的矩阵 ***T*** 转换成字符串矩阵 ***S***。

重要提示：矩阵 ***T*** 的元素一般要在 0～65535 之间，超出这个范围没有定义、但显示结果、系统会给出错误警告。

[例 2-20] 将正整数矩阵 ***T*** 转换成字符串矩阵，其中 ***T***=[102 67 132; 50 95 78]。

[算例代码]

```
%例 2-20
T =[102 67 132; 50 95 78];
S=char(T)
```

[运行结果]

```
S =

    fC

    2_N
```

（2）字符串转换方式二。

[函数命令]

int2str，num2str

[调用格式]

int2str(A)，num2str(A,k)，num2str(A,format)

int2str(A)将数或矩阵 ***A*** 转化为字符串或字符串矩阵；num2str(A,k)将数或矩阵 ***A*** 转化为字符串或字符串矩阵（最多 *k* 位有效位）。num2str(A, format)将数或矩阵 ***A*** 转化为字符串或字符串矩阵（格式输入参照 C 语言）。

[例 2-21]　将数值矩阵 ***A*** 转换成字符串矩阵，***A***=[1.2 6.7 3.2; 5.5 9.5 7.8]。

[算例代码]

```
%例 2-21
A =[1.2 6.7 3.2; 5.5 9.5 7.8];
B1= int2str(A)
B2=num2str(A,0)
B3=num2str(A, '%10.3f')                % 3 位有效位,10 位长
```

[运行结果]

```
B1 =

    1    7    3

    6   10    8

B2 =

    1          7          3

    6    1e+001           8

B3 =

    1.200      6.700      3.200

    5.500      9.500      7.800
```

[算例说明] 运行后得到的矩阵中数字已成为字符，***B***3 中输出是右对齐的。

（3）字符串转换方式三。

[函数命令]

eval，str2num

[调用格式]

eval(S)，str2num (S)

eval(S)表示将字符串 *S* 转化为数值；str2num (S)表示将字符串 *S* 转化为数值。

[例 2-22] 将字符串表达式 S ='a.*sin(w.*x)'转化为数值，其中 *a*=[1 2]，*w*=[1 3]，*x*=[1 2]

[算例代码]

```
%例 2-22
a=[1 2];
w=[1 3];
x=[1 2];
S ='a.*sin(w.*x)'
sinx=eval(S)
```

[运行结果]

```
S =
    a.*sin(w.*x)
sinx =
    0.8415    -0.5588
```

3．字符串比较

[函数命令]

strcmp

[调用格式]

strcmp(str1,str2)

strcmp(str1,str2)表示将两个字符串进行比较，相等时返回逻辑值为真。

[例 2-23] str1='bad'，str2='bad'，str3='dab'，判断它们是否相等。

[算例代码]

```
%例 2-23
str1='bad';
str2='bad';
str3='dab';
strcmp(str1,str2)
strcmp(str1,str3)
```

[运行结果]

```
ans =
    1
ans =
    0
```

4．字符串求值

[函数命令]

inline

[调用格式]

inline(exr)，inline(expr,arg1,arg2,…)

inline(exr)表示将字符串表达式转化为函数表达式；expr 为字符串表达式；arg1,arg2,…为

字符串，其作用是定义变量。

[例 2-24] 确定函数 $f = 5\sin(x) + 3\cos(y)$，x，y为自变量。

[算例代码]

```
%例 2-24
f=inline('5*sin(x)+3*cos(y)','x','y')
```

[运行结果]

```
f =
  Inline function:
  f(x,y) = 5*sin(x)+3*cos(y)
```

2.4　m 文件

m 文件是由 MATLAB 语言编写的、可在 MATLAB 语言环境下运行的程序源代码文件，它按 MATLAB 语言规则将命令及 MATLAB 内置函数有机地组合在一起，从而实现强大的功能。m 文件可以在 MATLAB 的程序编辑器中编写、也可以在文本编辑器中编写（后一种方式在编写大型程序时非常有用），都以“.m”为扩展名加以存储。MATLAB 语言中的 m 文件可以分为命令文件和函数文件两种。

2.4.1　命令文件

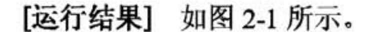
命令文件比函数文件简单，没有输入参数和输出参数、只是命令行的组合。命令文件可以对工作空间变量进行操作、也可以产生新变量。命令文件产生的所有变量都会保留在工作空间里，用户可以在以后的程序里对它进行操作，除非它们被相关的命令删除。命令文件调用的方法是，在 MATLAB 命令窗口直接输入文件名（filename），按 Enter 键即可。

重要提示：命令文件最好保存在 MATLAB 7.0\works 子目录（工作目录）下，文件名与内置函数及工具箱函数不应重名，与命令文件及工作空间中的变量也不应重名。有关文件操作的函数，如 fopen、close、fprintf、fscanf、fread，可利用 MATLAB 的帮助文件查看其用法。

[例 2-25] 分别绘制花瓣图案：$r = 2\sin^3(\theta)$，$r = \cos^3(\theta)$，$r = \sin^2(\theta)$，$r = 5\cos^3(3.5\theta)$。

[算例代码]

```
%例 2-25
clear;                                  %清除工作空间变量
theta=-pi:0.01:pi;
rho(1,:)=2*sin(5*theta).^2;             %计算命令
rho(2,:)=cos(10*theta).^3;
rho(3,:)=sin(theta).^2;
rho(4,:)=5*cos(3.5*theta).^3;
for i=1:4
subplot(2,2,i);
polar(theta,rho(i,:))                   %图形输出
end
```



[运行结果] 如图 2-1 所示。

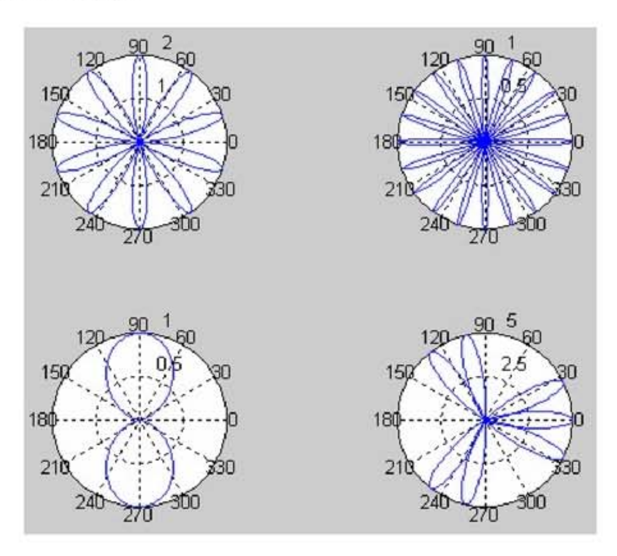


图 2-1　用命令文件绘制花瓣图案

2.4.2　函数文件

MATLAB 中的函数文件是使用扩展名为 m 的文件，它用来定义一个函数，定义过程中必须指定函数名和输入输出参数，并由 MATLAB 语句序列给出一系列的操作和处理，从而生成所需要的数据。函数文件是扩展 MATLAB 功能并对其进行二次开发的强有力工具。

函数文件格式一般包括以下部分。

- 函数定义行：函数定义行表明该 m 文件包含一个函数，并且定义函数名、输入和输出参数。
- 帮助信息第一行：在文件中的位置是第二行，这一行应该反映该 m 文件概括性的信息，lookfor（查找）命令只搜索和显示该行。
- 帮助正文：从第二行到第一非注释行之间的注释为帮助正文。对文件查询帮助信息时，将显示第二行和帮助正文。
- 函数体：函数体为所有计算过程和输入输出参数赋值的 MATLAB 代码，可以是调用函数、流程控制、交互式输入输出、计算、赋值、注释和空行等。
- 注释：注释语句以百分号（%）开头，可以出现在 m 文件的任何地方，也可以在一行代码的后面加注释语句，%后的代码部分为不执行部分。

函数的调用方式有以下两种。

（1）函数的嵌套调用：一个 m 函数调用其他函数称为函数的嵌套调用，被调用函数又可以调用其他函数，这称为函数的多层嵌套调用。

（2）函数的递归调用：在调用函数过程中，出现直接或间接调用函数本身的现象，称为函数的递归调用。递归调用函数中一定要有跳出递归调用的语句，否则函数会无穷循环下去。

重要提示：返回变量多于一个时应使用方括号；函数名与所存的 m 文件名应同名；输入变量多于一个时用逗号“,”隔开；注释说明是为了 help 命令的使用，但空行后不显示；函数

体中可使用错误提示信息：warning('message')。

同 C 语言一样，MATLAB 语言内置许多常用的数学函数，比如 sin、log、abs 等，它们的用法可通过查看帮助文件获得。

[例 2-26]　定义一个名为 f.m 的函数文件，并调用该函数。

[算例代码]

```
%例 2-26
a=[1 3];
b=[2 2];
c=f0226(a,b)

%定义 f0226.m 函数（下述代码另存为工作目录下的 f0226.m 文件）
function z=f0226(x,y)                          %在 m 文件中定义函数
z=sin(x.^2)+ y.^2-2*x-5;                       %函数表达式
```

[运行结果]

```
c =
  -2.1585    -6.5879
```

2.4.3　m 文件的调试

MATLAB 程序编辑器中提供了程序调试功能，下面介绍 m 文件调试的具体方法。

1．m 文件错误的种类

MATLAB 的 m 文件错误一般有语法错误和执行错误两种：语法错误发生在 m 文件程序代码的解释过程中，一般是函数参数输入类型有误或者矩阵运算阶数不符；执行错误则是程序运行过程中出现溢出或死循环等造成，执行错误与程序本身有关，并且较难发现和解决。

程序设计过程中应当避免出现 NaN、Inf 或空矩阵等，它们是程序运行时中最容易出现问题的地方。可在易于出现异常数值位置处使用控制语句来避免这一问题，识别函数有 isnan、isinf 及 isempty 等。

[例 2-27]　举例说明矩阵阶数不符的语法错误。

[算例代码]

```
%例 2-27
A=[1,2;3,4];
B=[1,2,3;4,5,6;7,8,9];
A*B
```

[运行结果]

```
???Error using= =>*
Inner matrix dimensions must agree.        %矩阵阶数不符
```

[例 2-28]　举例说明识别函数 isnan、isinf 与 isempty 的使用方法。

[算例代码]

```
%例 2-28
```

```
A=NaN;
B=[ ];
C=Inf;
isnan(A)
isinf(C)
isempty(B)
```

[运行结果]

```
ans =
     1
ans =
     1
ans =
     1
```

2．错误的识别

语法错误较好识别，因为 MATLAB 为方便用户的检查和定位会给出相应的错误信息。但执行错误较难识别，因为发生执行错误时，系统会结束对 m 文件的调用，这样将关闭函数的工作空间、无法获得需要的数据信息。MATLAB 提供了以下几种方法来获取所需要的中间信息。

- 将程序执行结果输出至命令窗口以检查运行的中间结果。方法是：把程序中的“;”去掉，增添变量的输出结果。
- 使用 keyboard 函数中断程序，使程序处于调试状态，命令窗口的提示符变为“K>>”，此时可以实现函数工作区间和命令窗口工作空间交互，从而获得所需要的信息。
- 将函数头注释掉，函数变成命令文件，操作对象相应变为命令窗口工作区间变量，从而获得所需要的信息。
- 使用调试菜单或调试函数。

3．调试过程

调试过程可通过调试菜单实现。这里介绍一些简单的调试函数，如表 2-8 所示。

表 2-8 MATLAB 语言中常用调试函数

调试函数	作　用
dbstop	用于在 m 文件中设置断点
dbstatus	显示断点信息
dbtype	显示 m 文件文本（包括行号）
dbstep	从断点处继续执行 m 文件
dbstack	显示 m 文件执行时调用的堆栈等
dbup/dbdown	实现工作空间的切换

2.5 程序设计

同其他程序设计语言一样，MATLAB 语言提供了丰富的流程控制语句进行具体的程序设计，MATLAB 语言的流程控制结构有顺序结构、分支结构和循环结构等三种。

2.5.1　顺序结构

顺序结构就是依次执行程序的各条语句。语句在程序文件中的物理位置反映了程序的执行顺序，一个典型的顺序结构，是不包含其他子结构和控制语句的批处理文件或是 MATLAB 中的命令文件。虽然大多数程序都包含子结构，但是它们整体上都是顺序结构。

[例 2-29]　举例说明顺序结构的使用方法。

[算例代码]

```
%例 2-29
disp ('the begin of the program')
disp ('the first line')
disp ('the end of the program')
```

[运行结果]

```
the begin of the program
the first line
the end of the program
```

2.5.2　分支结构

1．条件语句（if-else-end）

MATLAB 语言中的条件判断语句是 if-else-end 语句，该语句可以选择执行指定的命令。

[一般形式]

```
if  <逻辑判断语句>
    逻辑值为“真”时执行的语句
else
    逻辑值为“假”时执行的语句
end
```

重要提示：当逻辑判断表达式为“真”时，将执行 if 与 else 间的语句，否则将执行 else 与 end 间的语句。

[简化形式]

```
if  <逻辑判断语句>
    逻辑值为“真”时执行的语句
end
```

[嵌套形式]

```
if  <逻辑判断语句 1>
    逻辑值 1 为“真”时执行的语句
elseif  <逻辑判断语句 2>
    逻辑值 2 为“真”时执行的语句
elseif  <逻辑判断语句 3>
    逻辑值 3 为“真”时执行的语句
…
```

else
 当以上所有的逻辑值均为“假”时执行的语句
end

重要提示：在各层次的逻辑判断中，若其中任意一层逻辑判断为真，则将执行对应的执行语句，并跳出该条件判断语句，其后的逻辑判断语句均不进行检查。

[例 2-30] 举例说明条件语句一般形式的使用方法。

[算例代码]

```
%例 2-30
a=1;
if a==0;
    disp('nihao1');
else
    disp('nihao2');
end
```

[运行结果]

nihao2

[例 2-31] 举例说明条件语句简化形式的使用方法。

[算例代码]

```
%例 2-31
a=0;
if a==0;
    disp('nihao');
end;
```

[运行结果]

nihao

[例 2-32] 举例说明条件语句嵌套形式的使用方法。

[算例代码]

```
%例 2-32
a=0;
if a==0;
    disp('nihao1');
else if a==1;
        disp('nihao2');
else
        disp('nihao3');
end
end
```

[运行结果]

nihao1

2．开关语句（switch-case-end）

if-else-end 语句所对应的是多重判断选择，但有时也会遇到多分支判断选择的问题，MATLAB 语言为解决多分支判断选择提供了 switch-case-end 语句。与其他程序设计语言的 switch-case-end 语句不同，在 MATLAB 语言中，一个 case 语句后的条件为真时，switch-case-end 语句不对其后的 case 语句进行判断，即使有多条 case 判断语句为真，也只执行所遇到的第一条为真的语句，这样就不必像 C 语言那样，在每条 case 语句后加上 break 语句以防止继续执行后面为真的 case 条件语句。

[一般形式]

```
switch  <选择判断量>
case  选择判断值 1
          选择判断语句 1
case  选择判断值 2
          选择判断语句 2
…
otherwise
            判断执行语句
end
```

[例 2-33]　举例说明开关语句一般形式的使用方法。

[算例代码]

```
%例 2-33
a=2.5;
b=f0233(a)                              %调用函数 f0233.m
%定义 f0233.m 函数（下述代码另存为工作目录下的 f0233.m 文件）
function y=f0233(x)
switch x
case {1,2}
y=x*0.1;
case {3,4}
y=x*0.4;
otherwise
y=x*0.5;
end;
```

[运行结果]

```
 b=
    1.2500
```

2.5.3　循环结构

1．循环语句 1（for-end）

循环语句 for 是流程控制语句中的基础，可以用指定次数重复执行循环体内的语句。

[调用形式]

```
for  循环控制变量=<循环次数设定>
```

循环体

end

命令 for 和 end 中的字母必须小写，设定循环次数的数组可以是已定义数组、也可以在循环语句 for 中定义，定义格式为：

<初始值>：<步长>：<终值>

初始值为循环变量初始设定值，每执行循环体一次循环控制变量增加大小，循环控制变量值大于终值时循环结束，步长可以为负，循环体内不能对循环变量重新设置。此外，for 循环也允许嵌套使用。

[例 2-34] 举例说明循环语句 for 的使用方法。

[算例代码]

```
%例 2-34
for (i=1:3)
    for (j=1:4)
        A(i,j)=1.5;
    end;
end;
A
```

[运行结果]

```
A =
   1.5000    1.5000    1.5000    1.5000
   1.5000    1.5000    1.5000    1.5000
   1.5000    1.5000    1.5000    1.5000
```

2．循环语句 2（while-end）

while 循环语句与 for 循环语句不同的是，前者是以条件满足与否来判断循环是否结束，后者则是以执行次数是否达到指定值来判断。while 循环语句与 for 循环语句相同之处是两种语句均可实现循环执行，但 for 循环语句一般适用于已知循环次数、而不知道循环运算目标的问题，while 循环语句一般适用于已知循环运算目标、而循环次数未知的问题。

[一般形式]

while <循环判断语句>

循环体

end

重要提示：循环判断语句为逻辑判断表达式。当该表达式值为真时执行循环体内的语句；当表达式逻辑值为假时退出当前的循环体。

[例 2-35] 循环语句 while 的使用方法。

[算例代码]

```
%例 2-35
while (i==5)
    A(i)=1.5;
end;
A(i)
```

[运行结果]

```
ans =

     1.5000
```

3．循环语句的终止

在 while 循环语句中，语句内必须有修改循环控制变量的命令，否则该循环将陷入死循环，除非循环语句中有退出循环控制命令（如 break 语句）。当程序运行至退出循环控制命令时，不论循环控制变量是否满足循环判断语句，均将退出当前循环、执行循环后的其他语句。

与 break 语句对应，MATLAB 还提供了 continue 命令用于控制循环，当程序流程运行至命令时会忽略其后的循环体操作转而执行下一层次的循环。

习题 2

1．简述 MATLAB 中矩阵的建立、修改与引用方法。

2．简述 MATLAB 中函数与函数文件的建立、命名与引用方法，并以实例说明之。

3．已知 A(1,1)= '中国'，A(1,2)= '北京'，A(2,1)= '上海'， A(2,2)=[1 2 3]，试用 MATLAB 创建一个 2×2 的细胞数组 A。

4．已知学生有姓名、学号、性别、年龄、班级等信息，试用 MATLAB 创建相应的结构数组 student。

5．试用 MATLAB 的 inline 命令确定函数 $f=\sin(u)+\cos(v)$，这里，u，v 为自变量。

6．试用 MATLAB 创建一个关于变量 a，b 的函数，该函数返回 a，b 之差。

7．分别举例说明 MATLAB 中语句 if-else-end、switch-case-end、for-end、while-end 的用法。

8．请用中文在适当位置给下面的一段 MATLAB 程序添加注释。

```
function [x1,x2]=addplus(a,b)






x1=a+b;
x2=a*b;
```

第 3 章　矩阵线性代数算法实现

本章要点：

☑ MATLAB 中生成矩阵的几种方式；
☑ MATLAB 中矩阵的部分扩充、删除、修改及矩阵结构改变与变维的方法；
☑ MATLAB 中特殊矩阵函数及其生成方法；
☑ MATLAB 中矩阵基本运算（加减乘除、乘方、方阵运算、矩阵转置与矩阵函数）；
☑ MATLAB 中矩阵高级运算（求逆、范数、条件数、秩、分解算法及其线性方程组的各种解法）。

3.1　矩阵的生成

MATLAB 语言中，矩阵主要分为三类：数值矩阵、符号矩阵和特殊矩阵，其中数值矩阵有实数数值矩阵和复数数值矩阵两种。生成不同矩阵的方法并不完全相同，本节主要介绍生成实数数值矩阵的几种方法。

3.1.1　由命令窗口直接输入

MATLAB 语言的强大功能之一是能直接处理矩阵。当然首要任务是输入待处理的矩阵。任何矩阵都可以直接按行方式输入每个元素，输入时使用下述规则：同一行中不同元素用逗号（，）或用空格符来分隔、空格个数不限；不同行用分号（；）分隔或者分行输入；所有元素置于一对方括号（[ ]）内。

[例 3-1]　举例说明矩阵 $\boldsymbol{x}$ 的生成。

$$\boldsymbol{x}=\begin{matrix}1 & 2 & 3 & 4\\ 2 & 3 & 4 & 5\\ 3 & 4 & 5 & 6\end{matrix}$$

[算例代码]

```
%例 3-1
x = [1 2 3 4;2 3 4 5;3 4 5 6]
```

[运行结果]

```
x =
    1    2    3    4
    2    3    4    5
    3    4    5    6
```

3.1.2　由 m 文件生成

MATLAB 中的矩阵可在 m 文件中建立，在命令窗口中直接调用。对于大型矩阵，采用此方式更便于修改。

重要提示：m 文件中的变量名称与文件名不能相同，否则调用时会出现变量名与函数名的混乱。

[例 3-2]　用 m 文件建立大矩阵 ***x***，文件名为 abc.m。

```
x = [ 456    468    873     2    579    55
       21    687     54   488      8    13
       65   4567     88    98     21     5
      456     68   4589   654      5   987
     5488     10      9     6     33    77]
```

[算例代码]

```
%例 3-2
f0302
%定义 f0302 函数（下述代码另存为工作目录下的 f0302.m 文件）
x=[ 456   468   873     2   579    55
     21   687    54   488     8    13
     65  4567    88    98    21     5
    456    68  4589   654     5   987
   5488    10     9     6    33    77]
```

[运行结果]

```
x =
    456    468    873      2    579     55
     21    687     54    488      8     13
     65   4567     88     98     21      5
    456     68   4589    654      5    987
   5488     10      9      6     33     77
```

3.1.3　由文本文件生成

MATLAB 中的矩阵还可由文本文件生成，即在文件夹（通常为 work 文件夹）中建立 txt 文件，在命令窗口中直接调用此文件即可。

重要提示：txt 文件中不含变量名称，文件名 *x* 为矩阵变量名，每行数值个数必须相等。

[例 3-3]　用文本文件建立矩阵 ***x***，其中，

```
x=  1.1  1.2
    2.1  2.2
```

[算例代码]

```
%例 3-3
load f0303.txt
f0303
```

```
%定义 f0303.txt 文件（下述代码另存为工作目录下的 f0303.txt 文件）
1.1   1.2
2.1   2.2
```

[运行结果]

```
f0303 =
     1.1000     1.2000
     2.1000     2.2000
```

3.2 矩阵的修改

3.2.1 部分扩充

[调用格式]

D=[A;B C]

A 为原矩阵，***B***、***C*** 中包含要扩充的元素，***D*** 为扩充后的矩阵。

[例 3-4] 矩阵 ***A***=[1 2 3 4; 5 6 7 8]；***B***=eYe(2)；***C***=zeros(2)；扩充后 ***D***=[A;B C]，试求 ***D***。

[算例代码]

```
%例 3-4
A=[1 2 3 4; 5 6 7 8];
B=eYe(2);
C=zeros(2);
D=[A;B C]
```

[运行结果]

```
D =
   1     2     3     4
   5     6     7     8
   1     0     0     0
   0     1     0     0
```

3.2.2 部分删除

[调用格式]

A(:,n)=[];A(m,:)=[]

A(:,n)=[]表示删除矩阵 ***A*** 的第 *n* 列。A(m,:)=[]表示删除矩阵 ***A*** 的第 *m* 行。

[例 3-5] 删除矩阵 ***A*** 中的第二列，***A***=[1,2,3,4;5,6,7,8]。

[算例代码]

```
%例 3-5
A = [1 2 3 4;5 6 7 8];
A(:,2)=[]
```

[运行结果]

```
A =
  1    3    4
  5    7    8
```

3.2.3　部分修改

[调用格式]

A(m,n)=a; A(m,:)=[a b …]; A(:,n)=[a b …]

a、b、…为元素值；A(m,n)=a 表示修改矩阵 ***A*** 中第 *m* 行第 *n* 列元素为 *a*；A(m,:)=[a b …] 表示修改矩阵 ***A*** 中第 *m* 行的有元素为 *a*、*b*…；A(:,n)=[a b …]表示修改矩阵 ***A*** 中第 *n* 列的所有元素为 *a*、*b*…

重要提示：B=A(:)表示将矩阵 ***A*** 中的所有元素按行顺序合并成为一个向量。

[例 3-6]　修改矩阵 ***A*** 中的第二列元素为 10、11，其中，

A=1　2　3　4
　5　6　7　8

[算例代码]

```
%例 3-6
A = [1 2 3 4;5 6 7 8];
A(:,2)=[10 11]
```

[运行结果]

```
A =
  1   10    3    4
  5   11    7    8
```

3.2.4　结构改变

1．左右翻转

[函数命令]

fliplr

[调用格式]

fliplr(A)

fliplr(A)表示矩阵 ***A*** 行数不变，其元素左右翻转。

[例 3-7]　使矩阵 ***A*** 中的元素左右翻转，其中，

A=1　2　3　4
　5　6　7　8

[算例代码]

```
%例 3-7
A = [1 2 3 4;5 6 7 8];
fliplr(A)
```

[运行结果]

```
ans =
     4     3     2     1
     8     7     6     5
```

2．上下翻转

[函数命令]

flipud

[调用格式]

flipud(A)

flipud(A)表示矩阵 **A** 的行数保持不变，其元素上下翻转。

[例 3-8] 使矩阵 **A** 中的元素上下翻转。

```
A =1   2   3   4
    5   6   7   8
```

[算例代码]

```
%例 3-8
A = [1 2 3 4;5 6 7 8];
flipud(A)
```

[运行结果]

```
ans =
     5     6     7     8
     1     2     3     4
```

3．逆时针旋转

[函数命令]

rot90

[调用格式]

rot90(A); rot90(A,k);

rot90(A)表示矩阵 **A** 逆时针旋转 90 度，rot90(A,k)表示矩阵 **A** 逆时针旋转 $k\times 90$ 度，$k=\pm 1, \pm 2, \cdots$

[例 3-9] 使矩阵 **A** 中的元素左右翻转，**A**=[1,2,3,4;5,6,7,8]。

[算例代码]

```
%例 3-9
A = [1 2 3 4;5 6 7 8];
rot90(A)
```

[运行结果]

```
ans =
     4     8
     3     7
     2     6
     1     5
```

4．按指定维数翻转矩阵

[函数命令]

flipdim

[调用格式]

flipdim(A,dim)

flipdim(A,1) = flipud(A)，flipdim(A,2)=fliplr(A)。

[例 3-10]　使矩阵 ***A*** 中元素上下和左右翻转，***A***=[1,2,3,4;5,6,7,8]。

[算例代码]

```
%例 3-10
A = [1 2 3 4;5 6 7 8];
B1=flipdim(A,1)
B2=flipdim(A,2)
```

[运行结果]

```
B1 =
    5    6    7    8
    1    2    3    4
B2 =
    4    3    2    1
    8    7    6    5
```

5．平铺矩阵

[函数命令] repmat

[调用格式]

B = repmat(A,m,n); B= repmat(A,[m n]); B= repmat(A,[m n p…])

B = repmat(A,m,n)表示将矩阵 ***A*** 复制为 $m\times n$ 块，矩阵 ***B*** 由 $m\times n$ 块 ***A*** 平铺而成；B = repmat (A,[m n])与前面一致；B = repmat(A,[m n p…])表示矩阵 ***B*** 由 $m\times n\times p\times\cdots$ 个 ***A*** 块平铺而成。

重要提示：repmat(A,m,n)中 *A* 是一个数 *a* 时，该命令返回全由 *a* 组成的 $m\times n$ 矩阵。

[例 3-11]　将矩阵 ***A*** 平铺 3×2 块，***A***=[1,2;3,4]。

[算例代码]

```
%例 3-11
A = [1 2; 3 4];
B=repmat(A,3,2)
```

[运行结果]

```
B =
     1    2    1    2
     3    4    3    4
     1    2    1    2
     3    4    3    4
     1    2    1    2
     3    4    3    4
```

3.2.5 矩阵的变维

矩阵变维有两种方法，即使用“：”和函数 reshape。前者主要针对两个已知维数矩阵之间的变维操作；而后者是对于一个矩阵的操作。

1．使用“:”变维

[调用格式]

B(:)=A(:)

B(:)=A(:)表示经过 ***B*** 中元素与 ***A*** 中对应元素相乘后，得到新的结构不变的矩阵 ***B***。

[例 3-12] 若 ***A***=[1 2 5 4; 6 7 0 1]，***B***=ones(4,2)，求 B(:)=A(:)。

[算例代码]

```
%例 3-12
A=[1 2 5 4; 6 7 0 1]
B=ones(4,2)
B(:)=A(:)
```

[运行结果]

```
B =
    1    5
    6    0
    2    4
    7    1
```

2．使用 reshape 函数变维

[函数命令]

reshape

[调用格式]

B = reshape(A,m,n); B = reshape(A,m,n,p,⋯) ;

B = reshape(A,[m n p⋯]) ; B = reshape(A,siz)

B = reshape(A,m,n)表示返回以矩阵 ***A*** 元素构成的 $m\times n$ 维矩阵 ***B***; B = reshape (A,m,n,p,⋯)表示将矩阵 ***A*** 变维为 $m\times n\times p\times\cdots$；B = reshape(A,[m n p⋯])表示将矩阵 ***A*** 变维为 $m\times n\times p\times\cdots$；B = reshape(A,siz)表示 siz 决定变维大小，siz 是至少含有两个元素的向量。

重要提示：使用 reshape 命令时，***B*** 中元素个数与 ***A*** 相同。

[例 3-13] 已知 A=[1:8]，试用矩阵 ***A*** 元素构成 2×4 维的矩阵 ***B***。

[算例代码]

```
%例 3-13
A=[1:8];
B=reshape(A,2,4)
```

[运行结果]

```
B =
    1    3    5    7
    2    4    6    8
```

3.2.6　矩阵元素的数据变换

1．对由小数构成的矩阵 *A* 取整数

[函数命令]

floor; ceil; round; fix

[调用格式]

floor(A); ceil(A) ; round (A) ; fix (A)

floor(A)表示将矩阵 ***A*** 中元素按−∞方向取整，即取不足整数；ceil(A)表示将矩阵 ***A*** 中元素按+∞方向取整，即取过剩整数；round (A)表示将矩阵 ***A*** 中元素按最近整数取整，即四舍五入取整；fix (A)表示将矩阵 ***A*** 中元素按离 0 近的方向取整。

[例 3-14]　若 A=3*rand(2)，求 B1=floor(A)，B2=ceil(A)，B3=round(A)，B4=fix(A)。

[算例代码]

```
%例 3-14
A=3*rand(2)
B1=floor(A)
B2=ceil(A)
B3=round(A)
B4=fix(A)
```

[运行结果]

```
A =
    2.6739    1.3694
    2.2863    0.0555
B1 =
     2     1
     2     0
B2 =
     3     2
     3     1
B3 =
     3     1
     2     0
B4 =
    2     1
    2     0
```

结果中的 ***A*** 与随机数有关，用户所得结果可能与这里不尽相同。

2．矩阵的有理数形式

[函数命令]

rat

[调用格式]

[n,d]=rat (A)

[n,d]=rat (A)表示将矩阵 ***A*** 表示为两个整数矩阵相除，即：A=n./d。

[例 3-15] 已知矩阵 A=rand(2)，将矩阵 ***A*** 表示成两个整数矩阵相除。

[算例代码]

```
%例 3-15
[n,d]=rat (A)
```

[运行结果]

```
n =
    3575          367
    567           69
d =
    1337          268
    248         1243
```

由于 ***A*** 与随机数有关，用户所得结果可能与这里不尽相同。

3．矩阵元素的余数

[函数命令]

rem

[调用格式]

B = rem (A, x)

B = rem (A, x)表示矩阵 ***A*** 除以模数 ***x*** 后的余数。若 x=0，则定义 rem(A, 0) =NaN，若 $x \neq 0$，则整数部分由 fix(A./x)表示，余数 C=A−x.*fix (A./x)，允许模 ***x*** 为小数。

[例 3-16] 已知 A=rand(2)，求矩阵 ***A*** 除以模数 2 后的矩阵 ***B***。

[算例代码]

```
%例 3-16
B = rem (A, 2)
```

[运行结果]

```
B =
   0.6739    1.3694
   0.2863    0.0555
```

由于 ***A*** 与随机数有关，用户所得结果可能与这里不尽相同。

3.3 特殊矩阵

3.3.1 常用特殊矩阵函数

MATLAB 语言中内置了许多特殊矩阵的生成函数，通过这些函数可以自动生成一些具有某种特殊性质的矩阵，如表 3-1 所示。

表 3-1 MATLAB 语言的特殊矩阵

函 数 名	说 明
[]	生成空矩阵
eye	生成单位矩阵
zeros	生成零矩阵
ones	生成元素全为 1 的矩阵
rand	生成元素服从 0～1 之间随机分布的矩阵
randn	生成元素服从零均值单位方差正态分布的随机矩阵
diag	生成对角矩阵
triu	生成上三角矩阵
magic	生成魔方矩阵，各方向元素值和相等的方阵
hilb	生成 Hilbert 矩阵
toeplitz	生成 Toeplitz 矩阵
wilkinson	生成 Wilkinson 矩阵

3.3.2 特殊矩阵的生成方法

1．单位矩阵

[函数命令]

eye

[调用格式]

B=eye(n)；B=eye(m,n)；B=eye(size(A))

B=eye(n)表示生成 $n \times n$ 单位阵；B=eye(m,n)表示生成 $m \times n$ 单位阵；B=eye(size(A))表示生成与矩阵 $\boldsymbol{A}$ 相同大小的单位阵。

[例 3-17] 若 $\boldsymbol{A}$=[1 2 3; 2 3 4]，求 B=eye(size(A))。

[算例代码]

```
%例 3-17
A=[1 2 3; 2 3 4];
B=eye(size(A))
```

[运行结果]

```
B =
    1    0    0
    0    1    0
```

2．1 矩阵

[函数命令]

ones

[调用格式]

ones(n)；ones(m,n)；ones([m n])；ones(d1,d2,d3…)；

ones([d1 d2 d3…]) ones(size(A))

B=ones(n)表示生成 $n \times n$ 全 1 矩阵；B=ones(m,n)表示生成 $m \times n$ 全 1 矩阵；B=ones([m n])

表示生成 $m \times n$ 全 1 矩阵；B=ones(d1,d2,d3…)表示生成 $d1 \times d2 \times d3 \times \cdots$ 全 1 矩阵或数组；B=ones([d1 d2 d3…])表示生成 $d1 \times d2 \times d3 \times \cdots$ 全 1 矩阵或数组；B=ones(size(A))表示生成与矩阵 A 相同大小的全 1 矩阵。

[例 3-18] 若 ***A***=[1 2 3; 2 3 4]，求 B= ones(size(A))。

[算例代码]

```
%例 3-18
A=[1 2 3; 2 3 4];
B= ones(size(A))
```

[运行结果]

```
B =
     1     1     1
     1     1     1
```

3．零矩阵

[函数命令]

zeros

[调用格式]

zeros(n)；zeros(m,n)；zeros([m n])；zeros(d1,d2,d3…)；

zeros([d1 d2 d3…])； zeros(size(A))

B = zeros(n)表示生成 $n \times n$ 全零矩阵；B = zeros(m,n)表示生成 $m \times n$ 全零矩阵；B = zeros([m n])表示生成 $m \times n$ 全零矩阵；B = zeros(d1,d2,d3…)表示生成 $d1 \times d2 \times d3 \times \cdots$ 全零矩阵或数组；B = zeros([d1 d2 d3…])表示生成 $d1 \times d2 \times d3 \times \cdots$ 全零矩阵或数组；B = zeros(size(A))表示生成与矩阵 ***A*** 相同大小的全零矩阵。

[例 3-19] 若 ***A***=[1 2 3; 2 3 4]，求 B= zeros(size(A))。

[算例代码]

```
%例 3-19
A=[1 2 3; 2 3 4];
B=zeros(size(A))
```

[运行结果]

```
B =
     0     0     0
     0     0     0
```

4．随机矩阵

[函数命令]

rand

[调用格式]

rand；rand(n)；rand(m,n)；rand([m n])；rand(m,n,p,…)；

rand([m n p…])； rand(size(A))；

B = rand 表示无变量输入时只产生一个随机数；B= rand(n)表示生成 $n \times n$ 随机矩阵，其

元素在（0，1）内；B = rand(m,n)表示生成 $m\times n$ 随机矩阵；B = rand([m n])表示生成 $m\times n$ 随机矩阵；B = rand(m,n,p,…)表示生成 $m\times n\times p\times\cdots$ 随机矩阵或数组；B = rand([m n p…])表示生成 $m\times n\times p\times\cdots$ 随机矩阵或数组；B = rand(size(A))表示生成与矩阵 **A** 相同大小的随机矩阵。

[例 3-20]　若 **A**=[1 2 3; 2 3 4]，求 B= rand(size(A))。

[算例代码]

```
%例 3-20
A=[1 2 3; 2 3 4];
B= rand(size(A))
```

[运行结果]

```
B =
    0.2311    0.4860    0.7621
    0.6068    0.8913    0.4565
```

由于 **B** 与随机数有关，用户所得结果可能与这里不尽相同。

[例 3-21]　生成区间[10, 15]内均匀分布的 3 阶随机矩阵。

[算例代码]

```
%例 3-21
m=10;
n=15;
x=m+(n-m)*rand(3)
```

[运行结果]

```
x =
    10.0925    13.0772    13.6910
    14.1070    13.9597    10.8813
    12.2235    14.6091    12.0285
```

由于与随机数有关，用户所得结果可能与这里不尽相同。

[函数命令]

randn

[调用格式]

randn(n)；randn(m,n)；randn([m n])；randn(m,n,p,…)；

randn([m n p…])；　randn(size(A))。

B = randn(n)表示生成 $n\times n$ 正态分布随机矩阵；B = randn(m,n)表示生成 $m\times n$ 正态分布随机矩阵；B = randn([m n])表示生成 $m\times n$ 正态分布随机矩阵；B = randn(m,n,p,…)表示生成 $m\times n\times p\times\cdots$ 正态分布随机矩阵或数组；B = randn([m n p…])表示生成 $m\times n\times p\times\cdots$ 正态分布随机矩阵或数组；B = randn(size(A))表示生成与矩阵 **A** 相同大小的正态分布随机矩阵。

[例 3-22]　若 **A**=[1 2 3; 2 3 4]，求 B= randn(size(A))。

[算例代码]

```
%例 3-22
```

```
A=[1 2 3; 2 3 4]
B= rand(size(A))
```

[运行结果]

```
B =
    -0.4326    0.1253   -1.1465
    -1.6656    0.2877    1.1909
```

由于与随机数有关，用户所得结果可能与这里不尽相同。

[例 3-23] 生成均值为 0.5、方差为 0.05 的 3 阶矩阵。

[算例代码]

```
%例 3-23
mu=0.5;
sigma=0.05;
x=mu+sqrt(sigma)*randn(3)
```

[运行结果]

```
x =
     0.6322    0.2744    0.5000
     0.3561    0.4956    0.4289
     0.5850    0.4892    0.7449
```

由于与随机数有关，用户所得结果可能与这里不尽相同。

5．魔方矩阵

[函数命令]

magic

[调用格式]

M = magic(n)

M = magic(n)表示产生 n 阶魔方矩阵。

重要提示：魔方矩阵是指行、列、正、反斜对角线之和相等的矩阵，且 $n \neq 2$。

[例 3-24] 求 3 阶魔方矩阵。

[算例代码]

```
%例 3-24
M=magic(3)
```

[运行结果]

```
M =
    8    1    6
    3    5    7
    4    9    2
```

6．对角矩阵

[函数命令]

diag

[调用格式]

A = diag(v,k); A = diag(v); v = diag(A,k); v = diag(A)

A = diag(v,k)表示以向量 *v* 元素作为矩阵 ***A*** 的第 *k* 条对角线元素。当 *k*=0 时，*v* 为 ***A*** 的主对角线；当 *k*>0 时，*v* 为上方第 *k* 条对角线；当 *k*<0 时，*v* 为下方第 *k* 条对角线。

A = diag(v)表示以 *v* 为主对角线元素，其余元素为 0 构成 ***A***。

v = diag(A,k)表示抽取 ***A*** 的第 *k* 条对角线元素构成向量 *v*，*k*=0 时抽取主对角线元素；*k*>0 时抽取上方第 *k* 条对角线元素；*k*<0 时抽取下方第 *k* 条对角线元素。

v = diag(A)表示抽取主对角线元素构成向量 *v*。

[例 3-25]　已知 *v*=[1 2 3]，求 A1=diag(v,−1)，A2=diag(v,0)。

[算例代码]

```
%例 3-25
v=[1 2 3];
A1=diag(v,-1)
A2=diag(v,0)
```

[运行结果]

```
A1 =
    0    0    0    0
    1    0    0    0
    0    2    0    0
    0    0    3    0
A2 =
    1    0    0
    0    2    0
    0    0    3
```

7．三角矩阵

[函数命令]

tril; triu

[调用格式]

tril(A); L = tril(A,k) ;U = triu(A) ;U = triu(A,k)

L=tril(A)表示抽取矩阵 ***A*** 中主对角线的下三角部分构成矩阵 ***L***；L=tril(A,k)表示抽取矩阵 ***A*** 中第 *k* 条对角线的下三角部分（*k*=0 为主对角线，*k*>0 为主对角线以上，矩阵 *k*<0 为主对角线以下）；U = triu(A)表示抽取矩阵 ***A*** 中主对角线的上三角部分构成矩阵 ***U***；U = triu(A,k)表示抽取矩阵 ***A*** 中第 *k* 条对角线的上三角部分（*k*=0 为主对角线，*k*>0 为主对角线以上，*k*<0 为主对角线以下）。

[例 3-26]　已知矩阵 ***A*** 为 4 阶全 1 阵，求 L=tril(A,1)，U=triu(A,−1)。

[算例代码]

```
%例 3-26
A=ones(4);
L=tril(A,1)
```

```
U=triu(A,-1)
```

[运行结果]

```
L =
    1    1    0    0
    1    1    1    0
    1    1    1    1
    1    1    1    1
U =
    1    1    1    1
    1    1    1    1
    0    1    1    1
    0    0    1    1
```

8．Hilbert矩阵

[函数命令]

hilb

[调用格式]

H = hilb(n)

H = hilb(n)表示返回 n 阶 Hilbert 矩阵，其元素为 $H(i,j)=1/(i+j-1)$。

[例 3-27]　求一个 3 阶 Hilbert 矩阵。

[算例代码]

```
%例 3-27
H = hilb(3)
```

[运行结果]

```
H =
     1.0000    0.5000    0.3333
     0.5000    0.3333    0.2500
     0.3333    0.2500    0.2000
```

[函数命令]

invhilb

[调用格式]

H = invhilb(n)

H = invhilb(n)表示返回 n 阶逆 Hilbert 矩阵。

[例 3-28]　求一个 3 阶逆 Hilbert 矩阵。

[算例代码]

```
%例 3-28
H = invhilb(3)
```

[运行结果]

```
H =
```

```
     9    -36    30
   -36    192  -180
    30   -180   180
```

9．Toeplitz 矩阵

[函数命令]

toeplitz

[调用格式]

T = toeplitz(c,r)；T = toeplitz(r)

T = toeplitz(c,r)表示生成一个非对称的 Toeplitz 矩阵，将 *c* 作为第 1 列，将 *r* 作为第 1 行，其余元素与左上角相邻元素相等；T = toeplitz(r)表示用向量 *r* 生成一个对称的 Toeplitz 矩阵。

重要提示：当 *c* 与 *r* 中第一个元素冲突时，取 *c* 中第一个元素值，Toeplitz 矩阵不一定是方阵。

[例 3-29]　求一个 Toeplitz 矩阵。

[算例代码]

```
%例 3-29
c=[1 2 3 4 5];
r=[1.5 2.5 3.5 4.5 5.5];
T=toeplitz(c,r)
```

[运行结果]

```
T =
    1.0000    2.5000    3.5000    4.5000    5.5000
    2.0000    1.0000    2.5000    3.5000    4.5000
    3.0000    2.0000    1.0000    2.5000    3.5000
    4.0000    3.0000    2.0000    1.0000    2.5000
    5.0000    4.0000    3.0000    2.0000    1.0000
```

3.4　矩阵基本运算

3.4.1　加、减运算

[运算符] +、–

[运算规则]　对应元素相加、减，即按线性代数中矩阵的“+”，“–”运算进行。

[例 3-30]　$\boldsymbol{A}$=[1 2;3 4]，$\boldsymbol{B}$=[5 6 ;7 8]，求 $x1=\boldsymbol{A}+\boldsymbol{B}$，$x2=\boldsymbol{A}-\boldsymbol{B}$。

[算例代码]

```
%例 3-30
A=[1 2;3 4];
B=[5 6 ;7 8];
x1=A+B
x2=A-B
```

[运行结果]

```
x1 =
 6    8
10   12
x2 =
-4   -4
-4   -4
```

3.4.2 乘法运算

1．两个矩阵相乘

[运算符] *

[运算规则] 按线性代数中矩阵乘法运算进行，即前面矩阵各行元素分别与后面矩阵各列元素对应相乘并相加。

[例 3-31] ***A***=[1 2 3;4 5 6]，***B***=[5 6 ;7 8;9 10]，求矩阵 X=A*B。

[算例代码]

```
%例 3-31
A=[1 2 3;4 5 6];
B=[5 6 ;7 8;9 10];
X=A*B
```

[运行结果]

```
X =
     46    52
    109   124
```

2．矩阵的数乘

[运算符] *

[运算规则] 按线性代数中系数与矩阵相乘的法则进行，即系数分别与后面矩阵的各个元素相乘。

[例 3-32] 已知矩阵 ***A***=[1 2 3;4 5 6]，求矩阵 B=2*A。

[算例代码]

```
%例 3-32
A=[1 2 3;4 5 6];
B=2*A
```

[运行结果]

```
B =
   2    4    6
   8   10   12
```

3．矩阵点乘

[运算符] .*

[运算规则] 维数相同两个向量点乘，即向量对应元素相乘；或者维数相同两个矩阵对

应元素相乘。

[例 3-33]　***A***=[1 2 3; 4 5 6] ;***B***=[2 2 2;2 2 2]，求 C=A.*B。

[算例代码]

```
%例 3-33
A=[1 2 3; 4 5 6];
B=[2 2 2;2 2 2];
C=A.*B
```

[运行结果]

```
C =
    2     4     6
    8    10    12
```

4．内积

[函数命令]

dot

[调用格式]

dot(A,B)；dot(A,B,dim)

C = dot(A,B)表示 ***A***、***B*** 为向量时返回向量 *A* 与 *B* 的点积 *C*（*A* 与 *B* 长度相同）；***A***、***B*** 为矩阵时返回矩阵 ***A*** 与 ***B*** 的点积 ***C***（***A*** 与 ***B*** 具有相同维数）。C = dot(A,B,dim)表示在 dim 维数中给出 *A* 与 *B* 的点积。

重要提示：c= dot(a,b)相当于 c=sum(a.*b)。

[例 3-34]　***A***=[1 2 3; 4 5 6]；***B***=[2 2 2;2 2 2]，求 C= dot(A,B)。

[算例代码]

```
%例 3-34
A=[1 2 3; 4 5 6];
B=[2 2 2;2 2 2];
C= dot(A,B)
```

[运行结果]

```
C =
    10    14    18
```

[例 3-35]　***A***=[1 2 3; 4 5 6]；***B***=[2 2 2;2 2 2]，求 C= dot(A,B,2)。

[算例代码]

```
%例 3-35
A=[1 2 3; 4 5 6];
B=[2 2 2;2 2 2];
C= dot(A,B,2)
```

[运行结果]

```
C =
    12
```

30

[例 3-36] a=[−1 0 2];b=[−2 −1 1]，求 c= dot(a,b)。

[算例代码]

```
%例 3-36
a=[-1 0 3];
b=[-3 -1 1];
c=dot(a, b)
```

[运行结果]

```
c =
  6
```

5．叉积

数学上，两向量叉积是过两相交向量交点且垂直于两向量所在平面的向量。

[函数命令]

cross

[调用格式]

cross(A,B)；cross(A,B,dim)

C = cross(A,B)表示：若 A、B 为向量，则返回 A 与 B 的叉乘，即 $C=A\times B$，（A、B 必须是 3 个元素的向量）；若 $\boldsymbol{A}$、$\boldsymbol{B}$ 为矩阵，则返回一个 3×n 矩阵（列是 $\boldsymbol{A}$、$\boldsymbol{B}$ 对应列的叉积且 $\boldsymbol{A}$、$\boldsymbol{B}$ 都是 3×n 矩阵）。C = cross(A,B,dim)表示 dim 维中向量 A 与 B 的叉积，A 和 B 必须具有相同的维数，size(A,dim)和 size(B,dim)必须是 3。

[例 3-37] 已知向量 a=[1, 2, 3]和 b=[4, 5, 6]，求 c= cross(a,b)。

[算例代码]

```
%例 3-37
a=[1, 2, 3];
b=[4, 5, 6];
c= cross(a,b)
```

[运行结果]

```
c =
   −3     6    −3
```

6．混合积

混合积由点积和叉积两函数实现，使用时先叉乘后点乘，顺序不可颠倒。

[例 3-38] 向量 a=[1, 2, 3]，b=[4, 5, 6]，c=[7, 8, 9]，求三个向量的混合积 $X=a\cdot(b\times c)$。

[算例代码]

```
%例 3-38
a=[1, 2, 3];
b=[4, 5, 6];
c=[7, 8, 10];
X=dot(a, cross(b, c))
```

[运行结果]

```
X =

   -3
```

7．矩阵的卷积和多项式乘法

[函数命令]

conv

[调用格式]

conv(u,v)

w=conv(u,v)表示向量 u、v的卷积。数学上，长度为 m的向量序列 u和长度为 n的向量序列 v的卷积定义为：$w(k)=\sum_{j=1}^{k}u(j)v(k+1-j)$，式中，$w$向量序列的长度为$(m+n-1)$。

[例 3-39]　展开多项式$(s^2+2s+2)(s+4)(s+1)$。

[算例代码]

```
%例 3-39
w=conv([1,2,2],conv([1,4],[1,1]))          %求多项式系数向量 w
P=poly2str(w,'s')                          %将 w 表示成 s 的多项式
```

[运行结果]

```
w =

     1     7    16    18     8

P =

   s^4 + 7 s^3 + 16 s^2 + 18 s + 8
```

8．反褶积（解卷）和多项式除法运算

[函数命令]

deconv

[调用格式]

deconv(v,u)

[q,r]=deconv(v,u)表示多项式 v除以多项式 u，返回商多项式 q和余多项式 r。

重要提示：v、u、q、r都是按降幂排列的多项式系数向量。

[例 3-40]　求多项式$(x^3+2x^2+3x+4)(10x^2+20x+30)$的卷积。

[算例代码]

```
%例 3-40
u = [1 2 3 4];
v = [10 20 30];
c = conv(u,v)
[q,r] = deconv(c,u)
```

[运行结果]

```
c =

    10    40   100   160   170   120

q =
```

```
    10    20    30
r =
     0     0     0     0     0     0
```

9．张量积

[函数命令]

kron

[调用格式]

C=kron (A,B)

A 为 $m\times n$ 矩阵，***B*** 为 $p\times q$ 矩阵，则 ***C*** 为 $mp\times nq$ 矩阵。

重要提示：***A*** 与 ***B*** 的张量积定义为：$\boldsymbol{C}=\boldsymbol{A}\otimes\boldsymbol{B}=\begin{bmatrix} a_{11}\boldsymbol{B} & a_{12}\boldsymbol{B} & \cdots & a_{1n}\boldsymbol{B} \\ a_{21}\boldsymbol{B} & a_{22}\boldsymbol{B} & \cdots & a_{2n}\boldsymbol{B} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1}\boldsymbol{B} & a_{m2}\boldsymbol{B} & \cdots & a_{mn}\boldsymbol{B} \end{bmatrix}$，$\boldsymbol{A}\otimes\boldsymbol{B}$ 与 $\boldsymbol{B}\otimes\boldsymbol{A}$ 均为 $mp\times nq$ 矩阵，一般地 $\boldsymbol{A}\otimes\boldsymbol{B}\neq\boldsymbol{B}\otimes\boldsymbol{A}$。

[例 3-41] ***A***=[1 2;3 4]，***B***=[1 2 3;4 5 6;7 8 9]，求 ***C***=***A***⊗ ***B***。

[算例代码]

```
%例 3-41
A=[1 2;3 4];
B=[1 2 3;4 5 6;7 8 9];
C=kron(A,B)
```

[运行结果]

```
C =
     1     2     3     2     4     6
     4     5     6     8    10    12
     7     8     9    14    16    18
     3     6     9     4     8    12
    12    15    18    16    20    24
    21    24    27    28    32    36
```

3.4.3 除法运算

1．左除（\）和右除（/）

[运算符] “\” 和 “/”

重要提示：一般情况下，x=A\b 是方程 ***A*x*** =***b*** 的解，而 x=b/A 是方程 ***x*A***=***b*** 的解，如果 ***A*** 为非奇异矩阵，则 A*b* 和 *b*/A 可通过 ***A*** 的逆阵与 ***b*** 阵得到，A\b = inv(A)*b，b/A = b*inv(A)。

[例 3-42] ***A***=[1 0 3;4 13 6;7 4 9]，***b***=[4;7;1]，求 C=A\b。

[算例代码]

```
%例 3-42
A=[1 0 3;4 13 6;7 4 9];
```

```
b=[4;7;1];
C=A\b
```

[运行结果]

```
C =
   -3.1591
    0.4091
    2.3864
```

2．矩阵点除

[运算符] “./”

[基本格式] B./A

[运算法则] 相同维数的矩阵 ***A*** 和 ***B***, B./A 表示矩阵 ***B*** 中的元素除以矩阵 ***A*** 中的对应元素，结果矩阵的维数不变。

重要提示：若矩阵 ***A*** 和矩阵 ***B*** 中有一个为数，则此数与相应的每个元素作运算，结果矩阵与参加运算的矩阵维数相同。

[例 3-43] ***A***=[1 2 3;4 5 6]，***B***=[7 4 9;4 7 1]，求 C=B./A。

[算例代码]

```
%例 3-43
A=[1 2 3;4 5 6];
B=[7 4 9;4 7 1];
C=B./A
```

[运行结果]

```
C =
    7.0000    2.0000    3.0000
    1.0000    1.4000    0.1667
```

3.4.4　乘方运算

1．矩阵乘方

[运算符] “^”

[运算规则]

（1）当 ***A*** 为方阵、P 为大于 0 的整数时，A^P 表示 ***A*** 的 P 次方，即 ***A*** 自乘 P 次；P 为小于 0 的整数时，A^P 表示 $\boldsymbol{A}^{-1}$ 的$|P|$次方。

（2）当 ***A*** 为方阵，P 为非整数时，则 $\boldsymbol{A}\hat{}\ P = V\begin{bmatrix} d_{11}^{p} & & \\ & \ddots & \\ & & d_{nn}^{p} \end{bmatrix}V^{-1}$，其中 V 为 ***A*** 的特征向量，$\begin{bmatrix} d_{11} & & \\ & \ddots & \\ & & d_{nn} \end{bmatrix}$ 为特征值对角矩阵，如果有重根，以上指令不成立。

（3）标量的矩阵乘方 P^{A}定义为 $P^{A}=V\begin{bmatrix} p^{d_{11}} & & \\ & \ddots & \\ & & p^{d_{nn}} \end{bmatrix}V^{-1}$，式中 *V*，*D* 满足特征值分解 $AV=VD$。

[例 3-44] A=rand(3)，求 $B=A^2$。

[算例代码]

```
%例 3-44
A=rand(3);
B=A^2
```

[运行结果]

```
B =
    1.2921    1.2428    0.8176
    0.4369    0.9208    0.1372
    1.2512    1.6002    0.9658
```

由于与随机数有关，用户所得结果可能与这里不尽相同。

2．矩阵的数量乘方

[运算符] “.^”

[运算规则]

（1）矩阵的矩阵乘方 A.^B，表示矩阵 ***A*** 中元素对矩阵 ***B*** 中的对应元素求幂（***A***、***B*** 要为相同维数），结果矩阵与原矩阵维数相同。

（2）标量的矩阵乘方定义为 $P.\hat{}\ A=\begin{bmatrix} p^{a_{11}} & \cdots & p^{a_{1n}} \\ \vdots & & \vdots \\ p^{a_{m1}} & \cdots & p^{a_{mn}} \end{bmatrix}$。

（3）矩阵的标量乘方 A.^P，表示矩阵 ***A*** 中每个元素的 *P* 次乘方。

[例 3-45] A=rand(2)，B = [10;32]，求 $C=AB$。

[算例代码]

```
%例 3-45
A=rand(2);
B=[1 0;3 2];
C=A.^B
```

[运行结果]

```
C =
    0.4447    1.0000
    0.2331    0.8497
```

由于与随机数有关，用户所得结果可能与这里不尽相同。

3.4.5 矩阵函数

1．方阵的指数

[函数命令]

expm

[调用格式]

B = expm(A)；B = expmdemo1(A)；B = expmdemo2(A)；B = expmdemo3(A)

B=expm(A)表示使用 Pade 近似算法计算 e^{A}，A为方阵；B=expmdemo1(A)表示使用一个 m 文件和内部函数相同的算法计算 e^{A}；B=expmdemo2(A)表示使用泰勒级数计算 e^{A}；B=expmdemo3(A)表示使用特征值和特征向量计算 e^{A}。

[例 3-46]　A=[1 2;3 4]，求 $B=e^{A}$

[算例代码]

```
%例 3-46
A=[1 2;3 4];
B1=expm(A)
B2=expmdemo2(A)
```

[运行结果]

```
B1 =
     51.9690    74.7366
    112.1048   164.0738
B2 =
     51.9690    74.7366
    112.1048   164.0738
```

2．矩阵的对数

[函数命令]

logm

[调用格式]

B=logm(A)；[B,esterr]=logm(A)

B = logm(A)表示计算矩阵 A的对数，它是 expm(A)的反函数；esterr 为相对残差的估计值，即 norm(expm(B)–A)/norm(A)。

[例 3-47]　A=[1 1 0;0 0 2;0 0 –1]，B=expm(A)，求 C= logm(B)。

[算例代码]

```
%例 3-47
A=[1 1 0;0 0 2;0 0 -1];
B=expm(A)
C=logm(B)
```

[运行结果]

```
B =
    2.7183    1.7183    1.0862
         0    1.0000    1.2642
         0         0    0.3679
C =
    1.0000    1.0000    0.0000
```

```
        0        0    2.0000
        0        0   -1.0000
```

3．矩阵的函数

[函数命令]

funm

[调用格式]

F = funm(A,fun)；F = funm(A, fun, options)；

[F, exitflag] = funm(…)；

[F, exitflag, output] = funm(…)

F = funm(A,fun)计算由 fun 指定的 ***A*** 的矩阵函数，其中 ***A*** 为方阵，fun 可以是任意基本函数，如 sin、cos，等等，例如：funm(A, @exp)=expm(A)。

F = funm(A, fun, options)则通过 options 在结构选项中设置运算参数。

[例 3-48] 求矩阵函数 F=funm(magic(3), @sin)。

[算例代码]

```
%例 3-48
F=funm(magic(3), @sin)
```

[运行结果]

```
F =
    -0.3850     1.0191     0.0162
    0.6179      0.2168    -0.1844
    0.4173     -0.5856     0.8185
```

4．矩阵的方根

[函数命令]

sqrtm

[调用格式]

X = sqrtm(A)；[X,resnorm] = sqrtm(A)；[X,alpha,condest] = sqrtm(A)

X = sqrtm(A)表示 *X* 为矩阵 ***A*** 的平方根，即 ***X*** 满足 ***X**X*=*A***。若 ***A*** 的特征值有非负实部、则 *X* 是唯一的，若 ***A*** 的特征值有负的实部、则 ***X*** 为复矩阵，若 ***A*** 为奇异矩阵，则 *X* 不存在。

resnorm 是结果产生的相对误差。alpha 为稳定因子。condest 为结果条件数的估计值。

[例 3-49] 求 Y=sqrtm(X)，其中

```
X =[5   -4    1    0    0
   -4    6   -4    1    0
    1   -4    6   -4    1
    0    1   -4    6   -4
    0    0    1   -4    5]
```

[算例代码]

```
%例 3-49
X=[ 5 -4 1 0 0;-4 6 -4 1 0;1 -4 6 -4 1;0 1 -4 6 -4;0 0 1 -4 5 ];
Y=sqrtm(X)
```

[运行结果]

```
Y =
    2.0000   -1.0000    0.0000   -0.0000   -0.0000
   -1.0000    2.0000   -1.0000    0.0000   -0.0000
    0.0000   -1.0000    2.0000   -1.0000         0
   -0.0000    0.0000   -1.0000    2.0000   -1.0000
   -0.0000   -0.0000   -0.0000   -1.0000    2.0000
```

5．矩阵 *A* 的多项式

[函数命令]

polyvalm

[调用格式]

polyvalm(P, A)

P 为多项式系数向量；***A*** 为方阵，是多项式变量；polyvalm(P, A)返回多项式值。

[例 3-50]　*A*=[1 1 0;0 0 2;0 0 −1] , *P*=[1 2 3 4 5]，求 B=polyvalm(P, A)。

[算例代码]

```
%例 3-50
A=[1 1 0;0 0 2;0 0 -1];
P=[1 2 3 4 5];
B=polyvalm(P, A)
```

[运行结果]

```
B =
    15    10     8
     0     5     4
     0     0     3
```

3.4.6　矩阵转置

[运算符] “'”

[运算规则]　若矩阵 ***A*** 的元素为实数、则 A'返回 ***A*** 的转置；若 ***A*** 为复数矩阵，则 A'中的元素由 ***A*** 对应元素的共轭复数构成；若仅希望得到 ***A*** 的转置，则用命令：A.'。

[例 3-51]　A=magic(3)，求 B=A'。

[算例代码]

```
%例 3-51
A=magic(3)
B= A'
```

[运行结果]

```
A =
     8     1     6
     3     5     7
     4     9     2
B =
     8     3     4
     1     5     9
     6     7     2
```

3.4.7 方阵的运算

1．方阵的行列式

[函数命令]

det

[调用格式]

d=det(A)

d=det(A)返回方阵 ***A*** 的行列式的值。

[例 3-52] ***A***=[1 1 0;0 0 2;0 5 –1]，求 D= det(A)。

[算例代码]

```
%例 3-52
A=[1 1 0;0 0 2;0 5 -1];
D= det(A)
```

[运行结果]

```
D =
    -10
```

2．方阵的迹

[函数命令]

trace

[调用格式]

X=trace(A)

X=trace(A)返回矩阵 ***A*** 的行列式的迹，即对角线元素之和。

[例 3-53] ***A***=[1 1 0;0 0 2;0 5 –1]，求 X= trace (A)。

[算例代码]

```
%例 3-53
A=[1 1 0;0 0 2;0 5 -1];
X= trace (A)
```

[运行结果]

```
X =
     0
```

3.5　矩阵高级运算

3.5.1　矩阵的逆与伪逆

1．方阵的逆矩阵

[函数命令]

inv

[调用格式]

B=inv(A)

B=inv(A)表示求矩阵 ***A*** 的逆矩阵，若 ***A*** 为奇异阵或近似奇异阵、则给出警告信息。

[例 3-54]　***A***=[2 1 –1;2 1 2;1 –1 1]，求 B=inv(A)。

[算例代码]

```
%例 3-54
A=[2 1 -1;2 1 2;1 -1 1];
format rat                          %用有理格式输出
B=inv(A)
```

[运行结果]

```
B=
      1/3          0          1/3
      0            1/3        –2/3
      –1/3         1/3        0
```

2．矩阵的伪逆矩阵

[函数命令]

pinv

[调用格式]

B = pinv(A) ；B = pinv(A, tol)

B = pinv(A)表示求矩阵 ***A*** 的伪逆；tol 为误差。

重要提示：当矩阵为长矩阵时，方程 ***AX***=***I*** 和 ***XA***=***I*** 至少有一个无解，这时 ***A*** 的伪逆能在某种程度上代表矩阵的逆，若 ***A*** 为非奇异矩阵、则 pinv(A) = inv(A)。

[例 3-55]　已知 ***A*** 为 4 阶魔矩阵的前 3 列元素构成的矩阵，求 B=pinv(A)。

[算例代码]

```
%例 3-55
A=magic(4);          %产生 4 阶魔方矩阵
A=A(:,1:3);          %取 4 阶魔方矩阵的前 3 列元素构成矩阵 A
B=pinv(A)            %计算 A 的伪逆
```

[运行结果]

```
B =
     0.0522     0.0022     0.0272     – 0.0228
```

```
   -0.1831    0.2669    0.2919   -0.2581
    0.1603   -0.2397   -0.2897    0.3103
```

3.5.2 矩阵和向量的范数

1．向量的范数

[函数命令]

norm

[调用格式]

norm(X) ; norm(X,inf); norm(X,1); norm(X,-inf); norm(X, p)

n=norm(X)表示求欧几里德范数，即$\|X\|_2=\sqrt{\sum|x_k|^2}$，$X$为向量；n=norm(X,inf)表示求$\infty$-范数，即$\|X\|=\max(abs(X))$；n=norm(X,1)表示求 1-范数，即$\|X\|_1=\sum_k|x_k|$；n=norm(X,-inf)表示求向量 X中元素绝对值的最小值，即$\|X\|=\min(\mathrm{abs}(X))$；n=norm(X, p)表示求$p$-范数，即$\|X\|_p=\sqrt[p]{\sum_k|x_k|^p}$。

[例 3-56]　x= [0 1 2 3]，求 n1=norm(x), n2=norm(x,1), n3=norm(x,inf)。

[算例代码]

```
%例 3-56
x = [0 1 2 3];
n1=norm(x)
n2=norm(x,1)
n3=norm(x,inf)
```

[运行结果]

```
n1 =
    3.7417
n2 =
      6
n3 =
     3
```

2．矩阵的范数

[函数命令]

norm

[调用格式]

norm(A) ; norm(A,1); norm(A,2); norm(A,inf); norm(A, 'fro')

$\boldsymbol{A}$为矩阵；n = norm(A)表示求欧几里德范数$\|\boldsymbol{A}\|_2$、等于 $\boldsymbol{A}$的最大奇异值；n = norm(A,1)表示求矩阵 $\boldsymbol{A}$的列范数$\|\boldsymbol{A}\|_1$、等于 $\boldsymbol{A}$中列向量 1-范数的最大值；n = norm(A,2)表示求矩阵$\boldsymbol{A}$的欧几里德范数$\|\boldsymbol{A}\|_2$；n = norm(A,inf)表示求行范数$\|\boldsymbol{A}\|_\infty$、等于矩阵 $\boldsymbol{A}$中行向量 1-范数的最大值；n = norm(A, 'fro')表示求矩阵 $\boldsymbol{A}$的 Frobenius 范数$\|\boldsymbol{A}\|_F=\sqrt{\sum_i\sum_j|a_{ij}|^2}$。

[例 3-57]　***A***=[1 2 3;4 5 6]，求 n1=norm(A)，n2= norm(A,1)，n3= norm(A,inf)。

[算例代码]

```
%例 3-57
A= [1 2 3;4 5 6];
n1=norm(A)
n2=norm(A,1)
n3=norm(A,inf)
```

[运行结果]

```
n1 =
    9.5080
n2 =
         9
n3 =
        15
```

3.5.3　矩阵的条件数

[函数命令]

cond

[调用格式]

cond(A)；cond(A,p)

c=cond(A)求矩阵 ***A*** 的 2–范数的条件数；c = cond(A,p)表示求矩阵 ***A*** 的 ***p***–范数的条件数，***p*** 可以是 1、2、inf 或者'fro'。

重要提示：线性方程组 ***AX***=*b* 的条件数是大于或等于 1 的实数、用来衡量数据扰动、即 ***A*** 或 *b* 对解 ***X*** 的灵敏度，差条件方程组的条件数很大。

[例 3-58]　已知 ***A***=[1 2 3;4 5 6]，求 c=cond(A)。

[算例代码]

```
%例 3-58
A=[1 2 3;4 5 6];
c=cond(A)
```

[运行结果]

```
c =
    12.3022
```

3.5.4　矩阵的秩

[函数命令]

rank

[调用格式]

rank (A)；rank (A,tol)

k=rank (A)表示求矩阵 ***A*** 的秩；tol 为给定误差。

[例 3-59] 已知 ***A***=[1 2 3;4 5 6]，求 r=rank (A)。

[算例代码]

```
%例 3-59
A=[1 2 3;4 5 6];
r=rank (A)
```

[运行结果]

```
r =

    2
```

3.5.5 矩阵元素个数的确定

[函数命令]

numel

[调用格式]

numel(A)

n = numel(A)返回矩阵 ***A*** 中元素的个数 ***n***。

[例 3-60] ***A***=[1 2 3;4 5 6]，求 n= numel(A)。

[算例代码]

```
%例 3-60
A=[1 2 3;4 5 6];
n= numel(A)
```

[运行结果]

```
n =

    6
```

3.5.6 矩阵的分解

1．Cholesky 分解

[函数命令] chol

[调用格式] R = chol(X)；[R,p] = chol(X)

R = chol(X)返回满足 R'*R = X 的 ***R***。如果 ***X*** 为 ***n*** 阶对称正定矩阵，则 ***R*** 为实的非奇异上三角阵；若 ***X*** 非正定矩阵，则产生错误信息。

[R,p] = chol(X)返回满足 R'*R = X 的 ***R***、且不产生任何错误信息，若 ***X*** 为正定矩阵、则 ***p***=0；若 ***X*** 非正定矩阵、则 ***p*** 为正整数。

[例 3-61] 已知 X=pascal(3)，求[R,p]=chol(X)。

[算例代码]

```
%例 3-61
X=pascal(3);              %产生 3 阶 pascal 矩阵
[R,p]=chol(X)
```

[运行结果]

```
R =
     1     1     1
     0     1     2
     0     0     1
p =
     0
```

2．LU 分解

矩阵的三角分解又称 LU 分解，它的目的是将一个矩阵分解成一个下三角矩阵 ***L*** 和一个上三角矩阵 ***U*** 的乘积，即 ***A***=***LU***。

[函数命令]

lu

[调用格式]

[L,U] = lu(X)；[L,U,P] = lu(X)

[L,U] = lu(X)表示 ***U*** 为上三角阵，***L*** 为下三角阵或其变换形式，满足 ***LU***=***X***。

[L,U,P] = lu(X)表示 ***U*** 为上三角阵，***L*** 为下三角阵，***P*** 为单位矩阵的行变换矩阵，满足 ***LU***=***PX***。

[例 3-62]　***A***=[1 2 3;4 5 6;7 8 0]，求[L,U] = lu(A)。

[算例代码]

```
%例 3-62
A=[1 2 3;4 5 6;7 8 0];
[L,U] = lu(A)
```

[运行结果]

```
L =
    0.1429    1.0000         0
    0.5714    0.5000    1.0000
    1.0000         0         0
U =
    7.0000    8.0000         0
         0    0.8571    3.0000
         0         0    4.5000
```

3．QR 分解

矩阵的 QR 分解，是指将矩阵 ***A*** 分解成一个正交矩阵 ***Q*** 与一个上三角矩阵 ***R*** 的乘积。

[函数命令]

qr

[调用格式]

[Q,R] = qr(A)；[Q,R,E] = qr(A)；

[Q,R] = qr(A,0)；[Q,R,E] = qr(A,0)；

R=qr(A)；[C,R] = qr(A,b)；R = qr(A,0)；[C,R] = qr(A,b,0)

[Q,R] = qr(A)返回正交矩阵 ***Q*** 和上三角阵 ***R***，且满足 ***A***=***QR***。

[Q,R,E] = qr(A) 返回正交矩阵 ***Q***和上三角阵 ***R***，且满足 ***AE***=***QR***，***E*** 为单位矩阵的变换形式，***R***的对角线元素按大小降序排列。

[Q,R] = qr(A,0)表示产生矩阵 ***A***的"经济大小"分解。

[Q,R,E] = qr(A,0)中 ***E*** 的作用是使得 ***R*** 的对角线元素降序，且 Q*R=A(:, E)。

R=qr(A)返回稀疏矩阵 ***A***的分解，只产生一个上三角阵 ***R***，且满足 ***R****R*** = ***A'*A***。

[C,R] = qr(A,b)返回稀疏最小二乘问题 minimize||***Ax***–***b***||的两步解，[C,R] = qr(A,b)，x = R\c。

R = qr(A,0)返回稀疏矩阵 ***A***的经济型分解。

[C,R] = qr(A,b,0)返回稀疏最小二乘问题的经济型分解。

[例 3-63] ***A***=[1 2 3;4 5 6;7 8 0]，求[Q,R] = qr(A)。

[算例代码]

```
%例 3-63
A=[1 2 3;4 5 6;7 8 0];
[Q,R] = qr(A)
```

[运行结果]

```
Q =
    –0.1231    0.9045    0.4082
    –0.4924    0.3015   –0.8165
    –0.8616   –0.3015    0.4082
R =
    –8.1240   –9.6011   –3.3235
          0    0.9045    4.5227
          0         0   –3.6742
```

4．schur分解

[函数命令]

schur

[调用格式]

T = schur(A)；T = schur(A,flag)；[U,T] = schur(A,…)

T = schur(A)返回 schur 矩阵 ***T***，***T***的主对角线元素为特征值的三角阵。

Flag 为复特征根性质参数，若矩阵 ***A***有复特征根，则 flag='complex'，否则 flag='real'。

[U,T] = schur(A,…)返回正交矩阵 ***U***和 schur 矩阵 ***T***、且满足 ***A*** = ***U*T*U***。

[例 3-64] ***H*** = [–149 –50 –154; 537 180 546; –27 –9 –25]，求[U,T]=schur(H)。

[算例代码]

```
%例 3-64
H = [ -149   -50   -154; 537   180   546; -27   -9   -25 ];
[U,T]=schur(H)
```

[运行结果]

```
U =
      0.3162   – 0.6529    0.6882
```

```
  -0.9487   -0.2176    0.2294
   0.0000    0.7255    0.6882
T =
   1.0000   -7.1119  -815.8706
        0    2.0000   -55.0236
        0         0     3.0000
```

5．实 Schur 分解转化成复 Schur 分解

[函数命令]

rsf2csf

[调用格式]

[U,T] = rsf2csf (u,t)

[U,T] = rsf2csf (u,t)表示将实 Schur 形式转化成复 Schur 形式。

[例 3-65]　已知 ***A***=[1 2 3;4 5 6]，将满足 ***A***= ***U* T* U***的实 Schur 形式的 ***UT***转化为相应的复 Schur 形式。

[算例代码]

```
%例 3-65
A=[1 1 1 3;1 2 1 1;1 1 3 1;-2 1 1 4];
[u,t]=schur (A);
[U,T]=rsf2csf (u,t)
```

[运行结果]

```
U =
  -0.4916      -0.2756 - 0.4411i    0.2133 + 0.5699i   -0.3428
  -0.4980      -0.1012 + 0.2163i   -0.1046 + 0.2093i    0.8001
  -0.6751       0.1842 + 0.3860i   -0.1867 - 0.3808i   -0.4260
  -0.2337       0.2635 - 0.6481I    0.3134 - 0.5448i    0.2466
T =
   4.8121   -0.9697 + 1.0778i   -0.5212 + 2.0051i       -1.0067
        0    1.9202 + 1.4742i    2.3355             0.1117 + 1.6547i
        0         0              1.9202 - 1.4742i   0.8002 + 0.2310i
        0         0                   0                  1.3474
```

6．特征值分解

[函数命令]

eig

[调用格式]

d = eig(A)；d = eig(A,B)；[V,D] = eig(A)；

[V,D] = eig(A,'nobalance')；

[V,D] = eig(A,B)；[V,D] = eig(A,B,flag)

d = eig(A)返回矩阵 ***A***的特征值 ***d***，***d***以向量形式存放。

d = eig(A,B)返回广义特征值 ***d***（***A***、***B***为矩阵），***d***以向量形式存放。

[V,D] = eig(A)返回 ***A*** 的特征值对角阵 ***D*** 和特征向量 ***V***，使 ***AV***=***VD***。'nobalance'为误差调节作用参数。

[V,D] = eig(A,B)返回矩阵 ***A***、***B*** 的广义特征值向量阵 ***V*** 和广义特征值阵 ***D***，满足 ***V***=***BVD***。

flag 为算法参数，其可能值为：'chol' 表示对 ***B*** 使用 Cholesky 分解算法（***A*** 为对称 Hermitian 矩阵、***B*** 为正定阵）；'qz' 表示使用 QZ 算法（***A***、***B*** 为非对称或非 Hermitian 矩阵）。

[例 3-66] 已知 ***A***=[1 2 3;4 5 6;7 8 9]，求 d = eig(A), [V,D] = eig(A)。

[算例代码]

```
%例 3-66
A=[1 2 3;4 5 6;7 8 9];
d = eig(A)
[V,D] = eig(A)
```

[运行结果]

```
d =
    16.1168
    – 1.1168
    – 0.0000
V =
    – 0.2320    – 0.7858     0.4082
    – 0.5253    – 0.0868    – 0.8165
    – 0.8187     0.6123      0.4082
D =
    16.1168          0            0
          0    – 1.1168           0
          0          0     – 0.0000
```

7．奇异值分解

[函数命令]

svd

[调用格式]

S = svd (X)；[U,S,V] = svd (X)；[U,S,V] = svd (X,0)

S = svd (X)返回矩阵 ***X*** 的奇异值向量 ***S***。

[U,S,V] = svd (X)返回与 ***X*** 同大小的对角矩阵 ***S***、酉矩阵 ***U*** 和酉矩阵 ***V***，且满足 ***X***= ***U**** ***S**** ***V***，若 ***A*** 为 $m\times n$ 阵，则 ***U*** 为 $m\times m$ 阵、***V*** 为 $n\times n$ 阵，奇异值在 ***S*** 的对角线上，非负且按降序排列。

[U,S,V] = svd (X,0)返回“有效大小”分解，只计算出矩阵 ***U*** 的前 n 列，矩阵 ***S*** 的大小为 $n\times n$。

[例 3-67] ***A***=[1 2;3 4;5 6;7 8]，求[U,S,V]=svd(A)。

[算例代码]

```
%例 3-67
A=[1 2;3 4;5 6;7 8];
```

```
[U,S,V]=svd(A)
```

[运行结果]

```
U =
   – 0.1525   – 0.8226   – 0.3945   – 0.3800
   – 0.3499   – 0.4214    0.2428     0.8007
   – 0.5474   – 0.0201    0.6979   – 0.4614
   – 0.7448    0.3812   – 0.5462     0.0407
S =
   14.2691        0
         0   0.6268
         0        0
         0        0
V =
   – 0.6414    0.7672
   – 0.7672  – 0.6414
```

8．特征值问题的 QZ 分解

[函数命令]

qz

[调用格式]

[AA,BB,Q,Z]=qz(A,B);

[AA,BB,Q,Z,V,W]= qz(A,B);

qz(A,B,flag)。

A、***B*** 为矩阵；[AA,BB,Q,Z]=qz(A,B)返回上三角阵 ***AA*** 和 ***BB***、正交矩阵 ***Q*** 和 ***Z***，且满足 ***Q*A*Z***= ***AA*** 和 ***Q*B*Z***= BB；[AA,BB,Q,Z,V,W]= qz(A,B)同时返回列为特征向量的矩阵 ***V*** 和 ***W***；flag 为算法参数，其可能值为：'complex'表示复数分解（默认）；'real'表示实数分解。

[例 3-68]　*A*=[1 2 ;4 5]，B=pascal(2)，求[AA,BB,Q,Z,V,W]= qz(A,B)。

[算例代码]

```
%例 3-68
A=[1 2 ;4 5];
B=pascal(2);
[AA,BB,Q,Z,V,W]= qz(A,B)
```

[运行结果]

```
AA =
   – 0.5314    3.7211
          0    5.6455
BB =
    0.4079    0.9074
         0    2.4516
Q =
```

```
      0.6089    - 0.7933
    - 0.7933    - 0.6089
Z =
      0.8203    - 0.5719
    - 0.5719    - 0.8203
V =
      1.0000      0.2324
    - 0.6972    - 1.0000
W =
      1.0000    - 1.0000
    - 0.4343    - 0.7676
```

9．海森伯格形式的分解

如果矩阵 ***H*** 的第一子对角线下元素都是 0，则 ***H*** 为海森伯格(Hessenberg)矩阵，如果矩阵是对称矩阵，则它的海森伯格形式是对角三角阵。MATLAB 可以通过相似变换将矩阵变换成这种形式。

[函数命令]

hess

[调用格式]

H = hess(A)；[P,H] = hess(A)

H = hess(A)表示返回矩阵 ***A*** 的海森伯格形式；***P*** 为酉矩阵，满足 ***A*** = ***PHP***，且 P'P = eye(size(A))。

[例 3-69] ***A***=[−149 −50 −154;537 180 546; −27 −9 −25]，求[P,H]=hess(A)。

[算例代码]

```
%例 3-69
A=[-149 -50 -154;537 180 546;-27 -9 -25];
[P,H]=hess(A)
```

[运行结果]

```
P =
          1.0000          0          0
               0    -0.9987     0.0502
               0     0.0502     0.9987
H =
−149.0000     42.2037   −156.3165
−537.6783    152.5511   −554.9272
           0     0.0728      2.4489
```

3.6 求解线性方程组

线性方程组的求解可分为两类：一类是方程组求唯一解或求特解，另一类是方程组求无

穷解或通解，可以通过方程组系数矩阵的秩来判断：

（1）若系数矩阵的秩 $r=n$（n 为方程组中未知变量的个数），则有唯一解；

（2）若系数矩阵的秩 $r<n$，则可能有无穷解。

（3）线性方程组的无穷解 = 对应齐次方程组的通解+非齐次方程组的一个特解。

3.6.1　线性方程组唯一解或特解的解法

这类问题的解法有直接法和迭代法两种。直接法主要用于求解低阶稠密矩阵，迭代法常用于求解大型稀疏矩阵。

1．矩阵除法解法

[调用格式]

X=A\b。

$\boldsymbol{X}$方程组 $\boldsymbol{AX}=\boldsymbol{b}$ 的解答，$\boldsymbol{A}$ 为系数矩阵，$\boldsymbol{b}$ 为常数矩阵的转置。

[例 3-70]　求方程组 $\begin{cases} 5x_1+6x_2=1 \\ x_1+5x_2+6x_3=0 \\ x_2+5x_3+6x_4=0 \\ x_3+5x_4=0 \end{cases}$ 的解。

方法一：

[算例代码]

```
%例 3-70-1
A=[5   6   0   0;1   5   6   0;0   1   5   6;0   0   1   5];
b=[1 0 0 0]';
R_A=rank(A)                                                    %求秩
X=A\b                                                          %求解
```

[运行结果]

```
R_A =

        4

X =

     0.3081
    -0.0900
     0.0237
    -0.0047
```

方法二（函数 rref 求解）：

[算例代码]

```
%例 3-70-2
C=[A,b];                    %由系数矩阵和常数列构成增广矩阵 C
R=rref(C)                   %将 C 化成行最简行
```

[运行结果]

```
R =
```

```
    1.0000         0         0         0    0.3081
         0    1.0000         0         0   -0.0900
         0         0    1.0000         0    0.0237
         0         0         0    1.0000   -0.0047
```

则 R 的最后一列元素就是所求之解。

[例 3-71] 求方程组 $\begin{cases} x_1 + x_2 - 3x_3 - x_4 = 1 \\ 3x_1 - x_2 - 3x_3 + 4x_4 = 4 \\ x_1 + 5x_2 - 9x_3 - 8x_4 = 0 \end{cases}$ 的一个特解。

方法一：

[算例代码]

```
%例 3-71-1
A=[1 1 -3 -1;3 -1 -3 4;1 5 -9 -8];
b=[1   4   0]';
R_A=rank(A)
X=A\b
```

[运行结果]

```
R_A =

     2

Warning: Rank deficient, rank = 2,    tol =     8.8373e-015.
X =

         0
         0
   -0.5333
    0.6000
```

[算例说明] X=[0 0 −0.5333 0.6000]'是一个特解近似值，因矩阵 $\boldsymbol{A}$ 不满秩，此法得出的结果不够精确。

方法二（函数 rref 求解）：

[算例代码]

```
%例 3-71-2
A=[1 1 -3 -1;3 -1 -3 4;1 5 -9 -8];
b=[1   4   0]';
C=[A,b];                              %构成增广矩阵
R=rref(C)
```

[运行结果]

```
R =

    1.0000         0   -1.5000    0.7500    1.2500
         0    1.0000   -1.5000   -1.7500   -0.2500
         0         0         0         0         0
```

由此得解，向量 $\boldsymbol{X}$=[1.2500 −0.2500 0 0]'就是一个特解。

[算例说明]　若系数矩阵 ***A*** 不满秩、使用 rref 解法比较精确。

2．矩阵的 LU 、Q R 和 Cholesky 分解解法

（1）LU 分解

LU 分解又称 Gauss 消去分解，把矩阵分解为下三角矩阵的基本变换形式（行交换）和上三角矩阵的乘积，即 ***A***= ***LU***, ***L*** 为下三角矩阵，***U*** 为上三角矩阵，从而 ***A*X***= b 变成 ***L*U*X*** =b。

[调用格式]

X=U\(L\b)

[例 3-72]　求方程组 $\begin{cases} 4x_1 + 2x_2 - x_3 = 2 \\ 3x_1 - x_2 + 2x_3 = 10 \\ 11x_1 + 3x_2 = 8 \end{cases}$ 的一个特解。

[算例代码]

```
%例 3-72
A=[4 2 -1;3 -1 2;11 3 0];
b=[2 10 8]';
R_A=rank(A)
[L,U]=lu(A);
X=U\(L\b)
```

[运行结果]

```
R_A =

       2

Warning: Matrix is close to singular or badly scaled.
         Results may be inaccurate. RCOND = 2.018587e–017.

X =

   1.0e+016 *

   – 0.4053
     1.4862
     1.3511
```

[算例说明]　结果中的警告是由于系数矩阵不满秩造成的，可以通过 A*X 验证其正确性。

（2）Cholesky 分解

若 ***A*** 为对称正定矩阵，则 Cholesky 分解可将矩阵 ***A*** 分解成上三角矩阵和其转置的乘积，即 $A = R' * R$，其中 ***R*** 为上三角矩阵，同时，方程 $A*X=b$ 变成 $R' * R * X = b$。

[调用格式]

X = R\(R'\b)

[例 3-73]　求方程组 $\begin{cases} x_1 + x_2 + x_3 = 2 \\ x_1 + 2x_2 + 3x_3 = 10 \\ x_1 + 3x_2 + 6x_3 = 8 \end{cases}$ 的解。

[算例代码]

```
%例 3-73
A=[1 1 1;1 2 3;1 3 6];
b=[2 10 8]';
R=chol(A);
X=R\(R' \b)
```

[运行结果]

```
X =
    -16
     28
    -10
```

（3）QR 分解

对于任何长方矩阵 ***A***，都可以进行 QR 分解，其中 ***Q*** 为正交矩阵，***R*** 为上三角矩阵的初等变换形式，即 ***A***=***QR***，同时，方程 ***A********X***= *b* 变成 ***QRX***= *b*。

[调用格式]

```
X=R\(Q\b)
```

[例 3-74] 求方程组 $\begin{cases} x_1 + 2x_2 + 3x_3 = 2 \\ 4x_1 + 5x_2 + 6x_3 = 10 \\ 7x_1 + 8x_2 = 8 \end{cases}$ 的解。

[算例代码]

```
%例 3-74
A=[1 2 3;4 5 6;7 8 0];
B=[2 10 8]';
R_A=rank(A)
[Q, R]=qr(A);
X=R\(Q\B)
```

[运行结果]

```
R_A =
       3
X =
    4.4444
   -2.8889
    1.1111
```

重要提示：这三种分解（LU 分解、QR 分解和 cholesky 分解）的优点是运算速度快、可以节省磁盘空间、节省内存，在求解大型方程组时很有用。但当系数矩阵不满秩时，结果可能不准确。

3.6.2 齐次线性方程组通解的解法

MATLAB 中函数 null 用来求解零空间，即满足 ***A***·***X***=0 的解空间，实际上是求出解空间的一组基（基础解系），它可用于获得齐次线性方程组的通解。

[函数命令]

null

[调用格式]

z = null(A)； z = null(A,′r′)

z = null(A)返回 z，z的列向量为方程组的正交规范基，满足 $Z'\times Z=\boldsymbol{I}$； z = null(A,'r')返回 z，z的列向量是方程 $\boldsymbol{AX}$=0 的有理基。

[例 3-75]　求方程组 $\begin{cases} x_1+2x_2+2x_3+x_4=0 \\ 2x_1+x_2-2x_3-2x_4=0 \\ x_1-x_2-4x_3-3x_4=0 \end{cases}$ 的通解。

方法一：

[算例代码]

```
%例 3-75-1
A=[1  2  2  1;2  1  -2  -2;1  -1  -4  -3];
format  rat                              %指定有理数格式输出
B=null(A,'r');                           %求出解空间的有理基
syms  k1  k2                             %定义符号变量
X=k1*B(:,1)+k2*B(:,2)                    %写出方程组的通解
```

[运行结果]

```
X =
  2*k1+5/3*k2
 - 2*k1-4/3*k2
          k1
          k2
```

方法二：

[算例代码]

```
%例 3-75-2
A=[1  2  2  1;2  1  -2  -2;1  -1  -4  -3];
format  rat
B=rref(A)                              %通过行最简形得到基
```

[运行结果]

```
B =
     1      0     - 2    -5/3
     0      1       2     4/3
     0      0       0      0
```

所以原方程组的通解为 $X=k_1\begin{pmatrix}2\\-2\\1\\0\end{pmatrix}+k_2\begin{pmatrix}5/3\\-4/3\\0\\1\end{pmatrix}$

3.6.3 非齐次线性方程组通解的解法

1．一般解法

[基本步骤]

（1）判断 $\boldsymbol{AX}=\boldsymbol{b}$ 是否有解，若有解则进行下一步；

（2）求 $\boldsymbol{AX}=\boldsymbol{b}$ 的一个特解；

（3）求 $\boldsymbol{AX}=0$ 的通解；

（4）$\boldsymbol{AX}=\boldsymbol{b}$ 的通解为：$\boldsymbol{AX}=0$ 的通解+$\boldsymbol{AX}=\boldsymbol{b}$ 的一个特解。

[例 3-76] 求方程组 $\begin{cases} x_1 + x_2 - 3x_3 - x_4 = 1 \\ 3x_1 - x_2 - 3x_3 + 4x_4 = 4 \\ x_1 + 5x_2 - 9x_3 - 8x_4 = 0 \end{cases}$ 的通解。

[算例代码]

```
%例 3-76
A=[1   1   -3   -1;3   -1   -3   4;1   5   -9   -8];
b=[1   4   0]';
B=[A b];
n=4;
R_A=rank(A)
R_B=rank(B)
format rat
if R_A==R_B&R_A==n              %判断有唯一解
    X=A\b
elseif R_A==R_B&R_A<n           %判断有无穷解
    X=A\b                       %求特解
    C=null(A,'r')               %求 AX=0 的基础解系
else X='equition no solve'      %判断无解
end
```

[运行结果]

```
R_A =
      2
R_B =
      2
Warning: Rank deficient, rank = 2,   tol =    8.8373e－015.
X =
    0
    0
    －8/15
    3/5
C =
    3/2          －3/4
    3/2           7/4
```

```
    1         0
    0         1
```

所以原方程组的通解为 $X=k_1\begin{pmatrix}3/2\\3/2\\1\\0\end{pmatrix}+k_2\begin{pmatrix}-3/4\\7/4\\0\\1\end{pmatrix}+\begin{pmatrix}0\\0\\-8/15\\3/5\end{pmatrix}$

2．rref法

[例 3-77]　使用命令 rref 求方程组 $\begin{cases}x_1+x_2-3x_3-x_4=1\\3x_1-x_2-3x_3+4x_4=4\\x_1+5x_2-9x_3-8x_4=0\end{cases}$ 的通解。

[算例代码]

```
%例 3-77
A=[1   1   -3   -1;3   -1   -3   4;1   5   -9   -8];
b=[1 4 0]';
B=[A b];
C=rref(B)             %求增广矩阵的行最简形，可得最简同解方程组
```

[运行结果]

```
C =
    1        0       -3/2       3/4       5/4
    0        1       -3/2      - 7/4     -1/4
    0        0        0         0         0
```

对应齐次方程组的基础解系为：$\xi_1=\begin{pmatrix}3/2\\3/2\\1\\0\end{pmatrix}$，$\xi_2=\begin{pmatrix}-3/4\\7/4\\0\\1\end{pmatrix}$，非齐次方程组的特解为：$\eta^*=\begin{pmatrix}5/4\\-1/4\\0\\0\end{pmatrix}$，所以原方程组的通解为：$X=k_1\xi_1+k_2\xi_2+\eta^*$。

3.6.4　特殊线性方程组的解法

1．LQ 法解线性方程组

[适用对象]　线性方程组 $AX=b$ 中 A 为 n 阶对称方阵，b 为 n 元列向量。

[函数命令]

```
symmlq
```

[调用格式]

```
x=symmlq(A,b)
symmlq(A,b,tol)
symmlq(A,b,tol,maxit)
```

symmlq(A,b,tol,maxit,M)

symmlq(A,b,tol,maxit,M1,M2)

symmlq(A,b,tol,maxit,M1,M2,x0)

[x,flag] = symmlq(A,b,…)

[x,flag,relres] = symmlq(A,b,…)

[x,flag,relres,iter] = symmlq(A,b,…)

[x,flag,relres,iter,resvec] = symmlq(A,b,…)

[x,flag,relres,iter,resvec,resveccg] = symmlq(A,b,…)

其中，*x* 为 LQ 法对方程组 ***AX*=*b*** 的求解结果；maxit 为指定最大迭代次数；tol 为指定误差（默认值是 1e–6，即 10^{-6}）；maxit 为指定最大迭代次数；***M*** 为用于对称正定矩阵的预处理因子，***M***= ***M1***×***M2***；*x*0 为初始估计值（默认值为 0）；flag 的取值为如下：

- 0 表示在指定迭代次数之内按要求精度收敛；
- 1 表示在指定迭代次数内不收敛；
- 2 表示 ***M*** 为坏条件的预处理因子；
- 3 表示两次连续迭代完全相同；
- 4 表示标量参数太小或太大；
- 5 表示预处理因子不是对称正定的。

relres 表示相对误差。iter 表示计算 *x* 的迭代次数。resvec 表示每次迭代的残差。resveccg 表示每次迭代共轭梯度残差的范数。

重要提示：利用 LQ 法求解方程组 ***AX*=*b*** 时，*x* 收敛则显示结果信息，如不收敛则给出警告信息、相对残差、计算终止时的迭代次数，***A*** 可以是一个返回 ***A**X*** 的函数。

[例 3-78] 线性方程组 ***AX*=*b*** 中 ***A***=[1 3 0;3 4 8;0 8 5]，*b* = sum(A,2)，试用 LQ 法求 *x*。

[算例代码]

```
%例 3-78
A=[1 3 0;3 4 8;0 8 5];                         %产生一个对角矩阵
b = sum(A,2);
tol = 1e-8;                                    %指定精度
maxit = 15;                                    %指定最大迭代次数
[x,flag,relres,iter,resvec] = bicg(A,b,tol,maxit)
```

[运行结果]

```
x =
    1.0000
    1.0000
    1.0000
flag =
     0
relres =
       7.4955e–016
iter =
     3
```

```
resvec =
        20.2485
         1.9269
         1.6223
         0.0000
```

2．双共轭梯度法解方程组

[适用对象]　线性方程组 $AX=b$，A 为 n 阶对称方阵，b 为 n 元列向量。

[函数命令]

```
bicg
```

[调用格式]

```
x=bicg(A,b)
bicg(A,b,tol)
bicg(A,b,tol,maxit)
bicg(A,b,tol,maxit,M)
bicg(A,b,tol,maxit,M1,M2)
bicg(A,b,tol,maxit,M1,M2,x0)
[x,flag] = bicg(A,b,…)
[x,flag,relres] = bicg(A,b,…)
[x,flag,relres,iter] = bicg(A,b,…)
[x,flag,relres,iter,resvec] = bicg(A,b,…)
```

其中，x 为双共轭梯度法对方程组 $AX=b$ 的求解结果，其他符号意义与命令 symmlq 中的符号意义相同。

重要提示：x 收敛则显示结果信息，不收敛则给出警告信息、相对残差、计算终止的迭代次数，A 可以是一个返回 $A*X$ 的函数。

x=bicgstab(A,b)为稳定双共轭梯度法求解方程组 $AX=b$，调用方式、返回结果形式与命令 bicg 一样。

x=cgs(A,b)为复共轭梯度平方法求解方程组 $AX=b$，调用方式、返回结果形式与命令 bicg 一样。

x=lsqr(A,b)为共轭梯度 LSQR 方法求解方程组 $AX=b$，调用方式、返回结果形式与命令 bicg 一样。

x=pcg(A,b)为预处理共轭梯度方法求解方程组 $AX=b$，调用方式、返回结果形式与命令 bicg 一样，这里 A 为对称正定矩阵。

x = qmr(A,b)为准最小残差法求解方程组 $AX=b$，调用方式、返回结果形式与命令 bicg 一样，这里 A 为方阵。

x=minres(A,b)为最小残差法求解方程组 $AX=b$，这种方法是寻找最小残差来求 x，调用方式、返回结果形式与命令 bicg 一样，A 为对称矩阵。

[例 3-79]　线性方程组 $AX=b$，A 为一个对角矩阵，b=sum(A,2)，试用双共轭梯度法求 x。

[算例代码]

```
%例 3-79
```

```
n = 100;
on = ones(n,1);
A = spdiags([-2*on 4*on -on],-1:1,n,n);          %产生一个对角矩阵
b = sum(A,2);
tol = 1e-8;                                       %指定精度
maxit = 15;                                       %指定最大迭代次数
M1 = spdiags([on/(-2) on],-1:0,n,n);
M2 = spdiags([4*on -on],0:1,n,n);                 %M1*M2=M，即产生预处理因子
[x,flag,relres,iter,resvec] = bicg(A,b,tol,maxit,M1,M2,[])
```

[运行结果] x 的值全为 1。

```
flag =
        0
relres =
        5.2580e-009
iter =
        9
```

并且：resvec 略

3．广义最小残差法

[适用对象] 线性方程组 $AX=b$，A 为 n 阶方阵，b 为 n 元列向量。

[函数命令]

gmres

[调用格式]

```
x = gmres(A,b)
gmres(A,b,restart)
gmres(A,b,restart,tol)
gmres(A,b,restart,tol,maxit)
gmres(A,b,restart,tol,maxit,M)
gmres(A,b,restart,tol,maxit,M1,M2)
gmres(A,b,restart,tol,maxit,M1,M2,x0)
[x,flag] = gmres(A,b,…)
[x,flag,relres] = gmres(A,b,…)
[x,flag,relres,iter] = gmres(A,b,…)
[x,flag,relres,iter,resvec] = gmres(A,b,…)
```

其中，x 为广义最小残差法对方程组 $AX=b$ 的求解结果；restart 为指定迭代参数（默认值为系数方阵 A 的阶数 n），最大外部迭代值为 min(n/restart,10)、最大总迭代值为 restart*min(n/restart,10)；其他符号意义与命令 symmlq 中的符号意义相同。

[例3-80] 线性方程组 $AX=b$，其中 A=gallery('wilk',21)，b=sum(A,2)，试用广义最小残差法求 x。

[算例代码]

方法一：

```
%例 3-80-1
A = gallery('wilk',21);          %产生阶数为 21 的测试方阵
b = sum(A,2);
tol = 1e-12;
maxit = 15;
M1 = diag([10:-1:1 1 1:10]);
x = gmres(A,b,10,tol,maxit,M1,[],[])
```

[运行结果]　***x***全为 1。

方法二：

```
%例 3-80-2
A = gallery('wilk',21);
b = sum(A,2);
tol = 1e-12;
maxit = 15;
x = gmres(@afun,b,10,tol,maxit,@mfun,[],[],21)

%定义 afun 函数（下述代码另存为工作目录下的 afun.m 文件）
function y = afun(x,n)
y = [0; x(1:n-1)] + [((n-1)/2:-1:0)'; (1:(n-1)/2)'] .*x + [x(2:n); 0];

%定义 mfun 函数（下述代码另存为工作目录下的 mfun.m 文件）
function y = mfun(r,n)
y = r ./ [((n-1)/2:-1:1)'; 1; (1:(n-1)/2)'];
```

[运行结果]　***x***全为 1。

习题 3

1．分别用 m 文件和文本文件建立大矩阵 ***X***:

X=[1.23　2.34　3.45　4.56　5.67　6.78
　2.13　3.24　4.35　5.46　6.57　7.68
　3.42　4.53　5.64　6.75　7.86　8.97]

2．试在 MATLAB 中生成区间[20, 30]内均匀分布的 4 阶随机矩阵。

3．试用 MATLAB 分别生成 4 阶魔方矩阵和 4 阶 Hilbert 矩阵。

4．已知 ***A***=[1 2;3 4]，***B***=[2 3;4 5]，试用 MATLAB 分别计算 ***A*** 与 ***B*** 的乘积、点乘、点积、叉积。

5．已知 ***A***=[1 2;3 4]，试用 MATLAB 分别计算 ***A*** 的特征值对角阵、特征值、逆、转置、条件数、秩。

6．已知 ***A***=[1 2;3 4]，试用 MATLAB 分别计算 ***A*** 的 Cholesky 分解、LU 分解、QR 分解、schur 分解。

7．求解下列方程组。

（1）$\begin{cases} x_1 - 2x_2 + 3x_3 - 4x_4 = 1 \\ -2x_1 + 3x_2 - 4x_3 + 5x_4 = 2 \\ -3x_1 + 4x_2 + 5x_3 - 6x_4 = 3 \\ 4x_1 - 5x_2 - 6x_3 + 7x_4 = 4 \end{cases}$

（2）$\begin{cases} x_1 + 3x_2 + 5x_3 + 7x_4 = 12 \\ 2x_1 - 4x_2 + 6x_3 - 8x_4 = 13 \\ 3x_1 + 5x_2 - 7x_3 - 9x_4 = 14 \\ 4x_1 + 6x_2 + 8x_3 - 10x_4 = 15 \end{cases}$

（3）$\begin{cases} x_1 + x_2 + 3x_3 = 1 \\ x_2 - x_3 = 2 \\ x_1 + x_2 + 2x_3 = 3 \end{cases}$

（4）$\begin{cases} x_1 + x_2 + 3x_3 - x_4 = -2 \\ x_2 - x_3 + x_4 = 1 \\ x_1 + x_2 + 2x_3 + 2x_4 = 4 \\ x_1 - x_2 + x_3 - x_4 = 0 \end{cases}$

8．一段用 MATLAB 编制的 m 文件，运行时显示该程序出现如一些问题，请分析说明错误，并写出改正后的完整代码。

[m 文件]

```
clear;
a=[1 2 3 4;2 3 4 5;3 4 5 6];
b=[11 12 13 14;21 22 23 24;31 32 33 34];
c=a*b;
```

[错误信息]

??? Error using ==> *

Inner matrix dimensions must agree

第 4 章　数 据 处 理

本章要点：

- ☑ MATLAB 中一维、二维、多维数据插值的常用命令；
- ☑ MATLAB 中使用多项式和指定函数进行曲线拟合的方法；
- ☑ MATLAB 中数据统计分析处理的常用函数命令。

4.1　数据插值

MATLAB 中提供了包括使用多项式和快速 Fourier 变换的一维插值、使用不同插值方法的二维插值及多维插值，插值函数位于安装目录中子目录\toolbox\matlab\polyfun 之下。下面简述常用插值函数的使用方法及其典型算例。

4.1.1　一维插值

1．多项式插值

[函数命令]

interp1

[调用格式]

interp1(x,y,xi,method)

x 和 *y* 为既有数据的向量，其长度必须相同；*xi* 为要插值的数据点向量；method 为插值方法，可为“nearest”、“linear”、“cubic”、“spline”之一，分别为最近点插值、线性插值、分段三次 Hermite 插值和三次样条插值。

[例 4-1]　*x*，*y* 的实测数据见表 4-1，求 *x*1=2.55 处的插值结果。

表 4-1　例 4-1 的数据

x	1.0	2.0	3.0	4.0	5.0
y	11.2	16.5	20.4	26.3	30.5

[算例代码]

```
%例 4-1
x=[1.0 2.0 3.0 4.0 5.0];                    %输入变量数据 x
y=[11.2 16.5 20.4 26.3 30.5];               %输入变量数据 y
x1=2.55;                                    %输入待插值点 x
y11=interp1(x,y,x1,'nearest')               %最近点插值方法的插值结果
y12=interp1(x,y,x1,'linear')                %线性插值方法的插值结果
```

```
y13=interp1(x,y,x1,'cubic')                    %三次 Hermite 插值方法的插值结果
y14=interp1(x,y,x1,'spline')                   %样条插值方法的插值结果
```

[运行结果]

```
y11 =    20.4000
y12 =    18.6450
y13 =    18.6028
y14 =    18.4874
```

2．基于快速 Fourier 变换的 FFT 插值

[函数命令]

interpft

[调用格式]

y=interpft(x,n)

interpft(x,n)为通过内插得到的对 *x* 进行快速 Fourer 变换、长度 *n* 的向量，如果 ***x*** 是矩阵，则对每一列进行操作。

[例 4-2] 设 *x*=[1 2 3 4]，求对 *x* 进行 FFT 变换的内插结果。

[算例代码]

```
%例 4-2
x=[1 2 3 4];                                   %输入变量数据 x
y= interpft(x,6)                               %输出 FFT 变换结果 y
```

[运行结果]

```
y =    1.0000    1.3840    2.3840    3.0000    4.1160    3.1160
```

4.1.2 二维插值

[函数命令]

interp2

[调用格式]

interp2(x,y,z,xi,yi,method)

x，*y*，*z* 为既有数据的向量，其长度必须相同；*xi*，*yi* 为要插值的数据点；method 为插值方法，可为“nearest”、“linear”、“cubic”之一，分别为最近点插值、双线性插值、双三次插值。

[例 4-3] *x*，*y*，*z* 的实测数据如表 4-2 所示，求与 *x*1=2.3，*y*1=2.8 处的插值结果。

表 4-2 例 4-3 的数据

y/*x*	0.0	0.5	1.0	1.5	2.0	2.5	3.0
0.0	100	99	100	99	100	99	100
0.5	100	99	99	99	100	99	100
1.0	99	99	98	98	100	99	100
1.5	100	98	97	97	99	100	103
2.0	101	100	98	98	100	102	104
2.5	102	103	101	100	102	106	106
3.0	99	102	100	100	103	108	103

[算例代码]

```
%例 4-3
x=[0.0 0.5 1.0 1.5 2.0 2.5 3.0];                    %输入变量数据 x
y=[0.0 0.5 1.0 1.5 2.0 2.5 3.0];                    %输入变量数据 y
%输入变量数据 z
z=[100   99 100   99   100   99   100
100   99   99   99   100   99   100
99    99   98   98   100   99   100
100   98   97   97    99 100   103
101 100    98   98   100   102   104
102 103 101 100   102   106   106
99   102 100 100   103   108   103];
x1=2.3;y1=2.8;                                      %输入待插值点[x1,y1]
z11=interp2(x,y,z,x1,y1,'nearest')                  %最近点插值方法的插值结果
z12=interp2(x,y,z,x1,y1,'bilinear')                 %双线性插值插值方法的插值结果
z13=interp2(x,y,z,x1,y1,'bicubic')                  %双三次插值插值方法的插值结果
```

[运行结果]

```
z11 =   108
z12 =   105.3600
z13 =   105.9744
```

4.1.3 多维插值

[函数命令]

interp3,interpn

[调用格式]

interp3(x,y,z,xi,yi,zi,method), interpn(x,y,z,…,xi,yi,zi,…,method)

interp3 和 interpn 分别为三维插值和多维插值；*x*, *y*, *z*, …为既有数据；*xi*, *yi*, *zi*, …为需要插值的数据点；method 为插值方法，可为“nearest”、“linear”、“cubic”之一，分别为最近点插值、线性插值、三次插值。

4.2 曲线拟合

4.2.1 使用多项式曲线拟合

[函数命令]

polyfit

[调用格式]

p=polyfit(x,y,n), [p s]=polyfit(x,y,n), [p s mu]=polyfit(x,y,n)

p 为最小二乘意义上拟合多项式的相关系数；*x*, *y* 为数据点向量；*n* 为多项式阶次；*s* 为一结构，其包含了 Vandermonde 矩阵 ***R*** 的 Cholesky 因子、自由度 df、残差范数 normr；mu 为一向量，mu(1)、mu(2)分别为 *x* 的均值与标准差。

[函数命令]

polyval

[调用格式]

y=polyval(p,x)

y 为关于自变量 x、系数为 p 次多项式的值，x 为矩阵或向量时对其每一点进行操作；p 为一向量，各元素为多项式系数（降幂排列），对 n 次多项式来说，p 有 n+1 个元素，即 $y = p(1)*x$^$n + p(2)*x$^$(n-1)+\cdots+p(n)*x+p(n+1)$。

[例 4-4] 设 x=[0.0 0.3 0.8 1.1 1.6 2.3]'，y=[0.50 0.82 1.14 1.25 1.35 1.40]'，试求二次多项式拟合系数，并据此计算 $x1$=[0.9 1.2]时对应的 $y1$。

[算例代码]

```
%例 4-4
x=[0.0 0.3 0.8 1.1 1.6 2.3]';                %输入变量数据 x
y=[0.50 0.82 1.14 1.25 1.35 1.40]';          %输入变量数据 y
p=polyfit(x,y,2)                             %对 x，y 用二次多项式拟合，得到系数 p
x1=[0.9 1.2];                                %输入点 x1
y1=polyval(p,x1)                             %估计 x1 处对应的 y1
```

[运行结果]

```
p =   -0.2387    0.9191    0.5318
y1 =    1.1656    1.2909
```

4.2.2 使用指定函数进行曲线拟合

$\min\limits_{x}\left[\frac{1}{2}\sum\limits_{i} f_i^2(x_i)\right]$ 形式的最小平方问题（这里，$f_i(x)$为用户定义的函数，x 为自变量），实际上是对指定函数的曲线拟合，可使用优化工具箱中非线性曲线的拟合命令 lsqnonlin。

[函数命令]

lsqnonlin

[调用格式]

x=lsqnonlin(fun,x0)

x 为返回拟合参数；fun 为拟合曲线对应的函数，对应函数文件应保存在工作目录下、并针对所有数据点（因而输入数据在 fun 中定义）；$x0$ 为初始值。

[例 4-5] 试根据土的压缩试验数据（e–p 数据），用双曲线模型 $e = e_0 - \dfrac{p}{a+bp}$ 确定模型参数 a，b（压力 p=0 时的孔隙比，可用天然孔隙比 e_0 代替），已知 p=0，0.05，0.1，0.2，0.4，0.6，0.8，1.2 MPa 对应的孔隙比分别为 e=1.335，1.253，1.180，1.058，0.887，0.803，0.752，0.685。

[算例代码]

```
%例 4-5
Data =[0.0000 1.335;0.0500 1.253;0.1000 1.180;0.2000 1.058;0.4000 0.887;
0.6000 0.803;0.8000 0.752;1.2000 0.685];
```

```
p=Data(:,1);
e=Data(:,2);
plot(p,e,'g-');hold on;
x0=[1 0]';
x=lsqnonlin('f0405',x0)
e1=e(1);
e=e1-p./(x(1)+x(2)*p);
plot(p,e,'bo-');
%定义 f0405 函数（下述代码另存为工作目录下的 f0405.m 文件）
function f = f0405(x,Data)
Data =[0.0000 1.335;0.0500 1.253;0.1000 1.180;0.2000 1.058;0.4000 0.887;
0.6000 0.803;0.8000 0.752;1.2000 0.685];
p=Data(:,1);
e=Data(:,2);
e1=e(1);
z=e1-p./(x(1)+x(2)*p);
f = z-e;
```

[运行结果]

所得曲线拟合对比如图 4-1 所示，所得参数为 $x=[a\ b]=[x(1)\ x(2)]$，运行结果如下：

```
x =      0.4833      1.1104
```

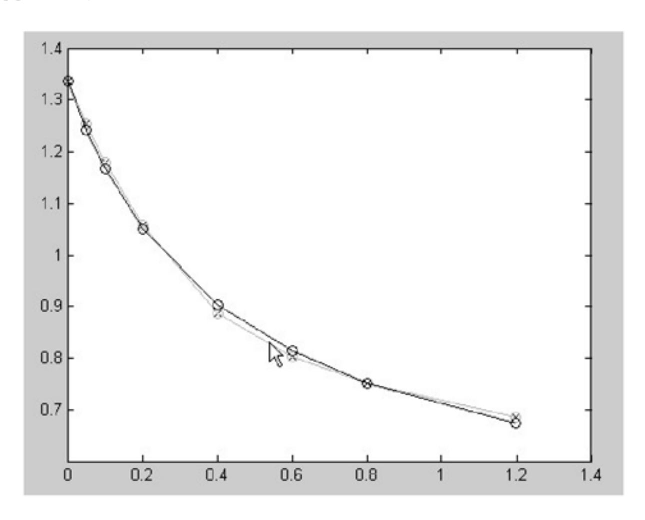


图 4-1　使用函数 lsqnonlin 对土压缩试验数据进行双曲线拟合

4.3　数据统计

MATLAB 中的数据统计分析函数位于安装目录中子目录\toolbox\matlab\datafun 之下，同时在优化工具箱、样条工具箱、统计工具箱等工具箱中提供了有关高级功能。下面通过典型算例及其注释说明 MATLAB 中数据统计分析的常用函数命令。

[例 4-6]　已知 ***A***=[1 3 7;8 1 5;6 9 1]，***B***=[10 1 8;5 7 10;9 1 7]，分别计算 ***A*** 中各列最大值及其对应行下标、(***A***,***B***)对应元素最大值、***A*** 中各列平均值、***A*** 中各列中值、***A*** 中各列标准差、***A*** 中各列元素之和、***A*** 中各列元素累计和、***A*** 中元素按列和行排序、***A*** 中各列协方差、(***A***,***B***)

相关系数、***A***中各列元素积与累计积。

[算例代码]

```
%例 4-6
A=[1 3 7;8 1 5;6 9 1];                    %输入已知数据 A
B=[10 1 8;5 7 10;9 1 7];                  %输入已知数据 B
maxa=max(A)                               %返回 A 列最大值
[maxa,I]=max(A)                           %返回 A 列最大值与对应行下标
maxab=max(A,B)                            %返回(A,B)对应元素最大值
meana=mean(A)                             %返回 A 列平均值
mediana=median(A)                         %返回 A 列中值
stda=std(A)                               %返回 A 列标准差
suma=sum(A)                               %返回 A 列元素之和
csuma=cumsum(A)                           %返回 A 列元素累计和
sorta1=sort(A)                            %返回 A 按列排序结果
sorta2=sortrows(A)                        %返回 A 按行排序结果
cova=cov(A',B')                           %返回 A 列向量协方差
corr=corrcoef(A',B')                      %返回 A',B'的相关系数
proda=prod(A)                             %返回 A 列元素积
cproda=cumprod(A)                         %返回 A 列元素累计积
```

[运行结果]

```
y11 =    20.4000
y12 =    18.6450
maxa =      8      9      7
maxa =      8      9      7
I =     2      3      1
maxab =
            10      3      8
             8      7     10
             9      9      7
meana =     5.0000    4.3333    4.3333
mediana =      6      3      5
stda =    3.6056    4.1633    3.0551
suma =     15     13     13
csuma =
             1      3      7
             9      4     12
            15     13     13
sorta1 =
             1      1      1
             6      3      5
             8      9      7
sorta2 =
```

```
          1     3     7
          6     9     1
          8     1     5
cova =
      10.0278   -3.5278
      -3.5278   12.0278
corr =
       1.0000   -0.3212
      -0.3212    1.0000
proda =    48    27    35
cproda =
          1     3     7
          8     3    35
         48    27    35
```

习题 4

1．已知 x=[0.0 0.3 0.8 1.1 1.6 2.3]，y=[0.40 0.56 1.14 1.31 1.56 2.10]，试用 MATLAB 对数据点对 x-y 进行二次多项式拟合，并使用不同插值方法计算 x=2 处对应的 y。

2．x，y，z 的实测数据如下表。试用 MATLAB 求 x_i=1.2、y_i=1.4 处的插值结果。

y/x	0.0	0.5	1.0	1.5
0.0	50	49	50	49
0.5	49	49	49	49
1.0	49	48	48	47
1.5	48	48	47	46

3．试用 MATLAB 求向量 y=[2 3 6 8]的最大值、最小值、均值、中值、标准差。

4．请以中文在本题适当位置给下面的一段 MATLAB 程序添加注释。

```
A=[1 3 7;8 1 5;6 9 1];
B=[10 1 8;5 7 10;9 1 7];
maxa=max(A);
maxab=max(A,B);
meana=mean(A);
mediana=median(A);
stda=std(A);
suma=sum(A);
cova=cov(A',B');
corr=corrcoef(A',B');
```

第 5 章　数 值 计 算

本章要点：

☑ MATLAB 中确定函数零点、单元函数和多元函数极值的方法；
☑ MATLAB 中计算数值积分的方法；
☑ MATLAB 中初值条件、边值条件问题的常微分方程求解方法；
☑ MATLAB 中常见偏微分方程的求解方法。

5.1　函数极值

在日常生活和工程实践中会碰到大量确定函数极值的问题，数学上通常通过求函数驻点（导数为 0 的点）的方法来获得函数的极大值和极小值，当函数导数不存在或者导数存在但导数比较复杂时，用数值方法求函数极值就快捷得多。下面介绍使用 MATLAB 确定一元函数与多元函数极值的方法。

5.1.1　一元函数的极值

[函数命令]

```
fminbnd
```

[调用格式]

```
[x,feval,exitflag,output]=fminbnd(fun,x1,x2,options,P1,P2,…)
```

x 为函数 fun 在区间 $x1<x<x2$ 上的极值。

feval 为求得极值时的函数值。

exitflag 为收敛描述标志：

- exitfalg＞0，函数收敛到 *x*；
- exitfalg＝0，函数计算次数已达最大迭代次数；
- exitfalg＜0，函数在计算区间内不收敛。

output 为包含最优化信息的结构，output.algorithm、output.funccount、output.iterations 分别为所用算法、函数迭代次数、所用迭代次数。

*x*1，*x*2 为极值点所在区间。

options 为可选项，可以选择 optimset 函数给出的 18 个选项中的 4 个。

- Display 为显示层次：off 时不显示输出内容；iter 时显示迭代过程；final 时显示输出内容；notify 时函数不收敛则显示输出（默认选择）。
- MaxFunEvals 为函数值最大误差。
- MaxIter 为迭代最大步数（默认值为 500）。

➢ TolX 为 x 误差范围（默认值为 1.0e−4）。

$P1,P2,\ldots$为函数 fun 的附加参数。

[例 5-1]　求函数 $f(x)=x^3-2x-5$ 在区间(0,2)上的极值。

[程序代码]

```
%例 5-1
f=inline('x.^3-2*x-5');                     %通过内联函数建立函数 f
x1=fminbnd(f,0,2)                           %求函数 f 在区间(0,2)上的极值
x2=fminbnd(f,0,2,optimset('TolX',1e-12,'Display','off'))
                                            %求函数 f 在区间(0,2)上的极值并进行有关计算设置
[x3,fx,out]=fminbnd(f,0,2)                  %fx 为返回输出函数值
                                            %out 为返回输出迭代次数，计算次数，所用算法
```

[运行结果]

x1 =0.8165

x2 =0.8165

x3 =0.8165

fx =-6.0887

out =1

5.1.2　多元函数的极值

[函数命令]

fminsearch

[调用格式]

[x,feval,exitflag,output]= fminsearch (fun,x0,options,P1,P2,⋯)

x 为函数 fun 在 $x0$ 附近的极值；$x0$ 为 x 的初始值；其他符号的意义与求一元函数极值时的意义相同。

[例 5-2]　求函数 $f(x,y,z)=x^2+2.5\sin y-z^2x^2y^2$ 的极值，初值为 $x_0=-0.6$，$y_0=-1.2$，$z_0=0.135$.

[程序代码]

```
%例 5-2
[xx,fx,out]=fminsearch(@f0502,[-0.6 -1.2 0.135])
%定义 f0502 函数（下述代码另存为工作目录下的 f0502.m 文件）
function y1= f0502(v)
x=v(1);
y=v(2);
z=v(3);
y1=x.^2+2.5*sin(y)-z.^2*x.^2*y.^2;
```

[运行结果]

xx =0.0000　−1.5708　0.1803

fx =−2.5000

out =1

5.2 函数零点

[函数命令]

fzero

[调用格式]

[x,feval,exitflag,output]= fzero(fun,x0,options,P1,P2,…)

x 为函数 fun 在 *x*0 附近的零点；*x*0 为 *x* 的初始值；其他符号的意义与求一元函数极值时的意义相同。

[例 5-3] 求 $f(x)=x^2-5x+6$ 的零点，初值为 $x_0=0.0$。

[程序代码]

```
%例 5-3-1
[xx,fx,out]=fzero('x.*x-5*x+6',0.0)
```

[运行结果]

xx =2.0000

fx =0

out =1

函数 $f(x)=x^2-5x+6$ 在 $x_0=0.0$ 附近的一个零点是 2.0，它还有另外一个零点 3.0，后者可以通过改变初始值 x_0 达到，执行下面的命令可以得到另外一个零点：

```
%例 5-3-2
[xx,fx,out]=fzero('x.*x-5*x+6',4.0)
```

[运行结果]

xx =3.0000

fx =0

out =1

5.3 数值积分

[函数命令]

quad,quadl,odblquad

[调用格式]

q1=quad(fun,x1,x2,tol,trace,p1,p2…);

q2=quadl(fun,x1,x2,tol,trace,p1,p2…);

q3=dblquad(fun,xmin,xmax,ymin,ymax,tol,method,p1,p2…);

*q*1 为自适应 Simpson 方法（默认方法）的积分结果，*q*2 为自适应 Lobatto 方法的积分结果，*q*2 为二重积分结果，quad、quadl、odblquad 为三者对应的函数命令；fun 为被积函数；*x*1、*x*2 分别为积分下限和上限；tol 为计算结束时的相对误差；trace 为显示中间计算结果参数，默认值为 0（不显示中间结果）；*p*1,*p*2,…为未知参数。*x*min、*x*max 分别为二重积分时自

变量 x 的下限和上限；ymin、ymax 分别为二重积分时自变量 y 的下限和上限；method 为积分方法。

[例 5-4]　求积分 $\int_0^{3\pi}\sqrt{4\cos^2(2t)+\sin^2 t}\,\mathrm{d}t$。

[算例代码]

```
%例 5-4
Q1=quad('sqrt(4*cos(2*t).^2+sin(t).^2+1)',0,3*pi)          %Simpson 方法计算结果
Q2=quadl('sqrt(4*cos(2*t).^2+sin(t).^2+1)',0,3*pi)         %Lobatto 方法计算结果
```

[运行结果]

Q1 = 17.2220

Q2 = 17.2220

[例 5-5]　求二重积分 $\int_{\pi}^{2\pi}\int_{3\pi}^{4\pi}(y\sin x+x\sin y)\,\mathrm{d}x\mathrm{d}y$。

[程序代码]

```
%例 5-5
Q3=dblquad('y*sin(x)+x*cos(y)',3*pi,4*pi,pi,2*pi)
```

[运行结果]

Q3 =　−29.6088

5.4　求解常微分方程

5.4.1　带初值条件的常微分方程

常微分方程，即 Ordinary Differential Equation，简称 ODE，带初值条件的常微分方程是指表达形式为 $\begin{cases} y'=f(t,y) \\ y(t_0)=y_0 \end{cases}$ 的方程组，MATLAB 中用积分器 solver 来求解；对于高阶方程 $y^{(n)}=f(t,y,y',\cdots,y^{(n-1)})$，通常解法是假设 $y_1=y$，从而 $y_1=y$，$y_2=y'$，…，$y_n=y^{(n-1)}$，于是高阶方程就可化为下述常微分方程组求解：

$$\begin{cases} y_1'=y_2 \\ y_2'=y_3 \\ \vdots \\ y_n'=f(t,y,y',\cdots,y^{(n-1)}) \end{cases}$$

[函数命令]

solver

[调用格式]

[T,Y]=solver(odefun,tspan,y0,options,p1,p2…)

T，*Y* 为微分方程组的解；solver 为积分器，可为 ode45、ode23、ode113、ode15s、ode23s、

ode23t、ode23tb 之一，它们对应不同的积分方法，常用的积分器是 ode45（Runge-Kutta 方法）；odefun 为描述微分方程组的函数；otspan 为指定积分区间的向量；y0 为初始条件向量；options 为 odeset 命令设定的可选函数，可采用系统默认值；p1，p2，…为未知参数。

[例 5-6] 求二阶 Van der Pol 方程 $y''-\mu(1-y^2)y'+y=0$ 在 $\mu=20$、$y(1)=2$、$y(2)=0$ 时的数值解。

[解题过程] 使用常用方法改变方程形式，令 $y_1=y$，$y_2=y_1'$，则 $y_1'=y_2$，$y_2'=\mu(1-y_1^2)y_2-y_1$；将其写为函数文件 vdpl.m；使用命令 ode45 确定 x、y，并作出 y x的关系曲线图。

[算例代码]

```
%例 5-6
mu=20;
%对参数 mu 在区间[]0 20 求函数 vdpl 在 y(1)=2 与 y(2)=0 时的积分结果
[x,y]=ode45(@f0506,[0 20],[2;0],mu);
plot(x,y);                                   %画出 y  x 的图形
%定义 f0506 函数（下述代码另存为工作目录下的 f0506.m 文件）
function y= f0506(x,y);                              %函数定义
y=[y(2);(1-y(1)^2)*y(2)-y(1)];
```

[运行结果] 运行结果见图 5-1。

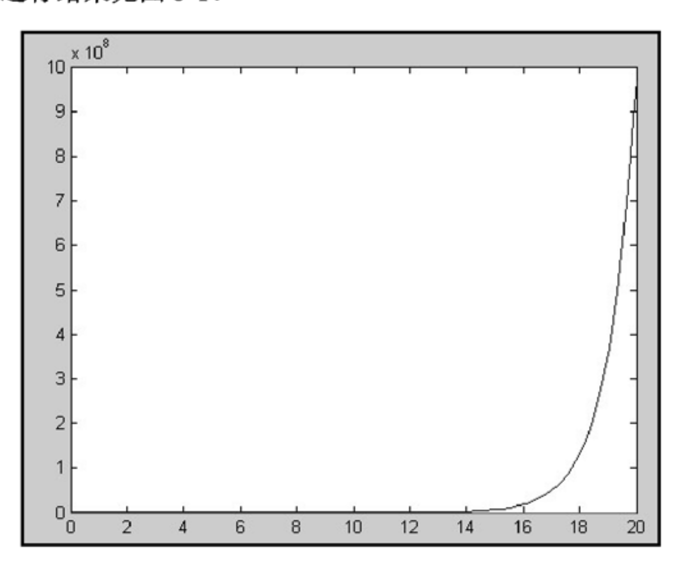


图 5-1 带初值条件微分方程 $y''-\mu(1-y^2)y'+y=0$ 的数值解

5.4.2 带边值条件的常微分方程

带边值条件的常微分方程，即 boundary value problems，简称 BVP 问题，是指表达形式为 $\begin{cases} y'=f(x,y) \\ g(f(a),f(b))=0 \end{cases}$ 或 $\begin{cases} y'=f(x,y,p) \\ g(f(a),f(b),p)=0 \end{cases}$ 的方程组（p 是未知参数），在 MATLAB 中使用积分器 bvp4c 来求解。

[函数命令]

bvp4c

[调用格式]

sol=bvp4c(odefun,bcfun,solinit,options,p1,p2,⋯)

sol 为一结构，sol.x、sol.y、sol.yp 分别是所选择的网格点及其对应的 $y(x)$与 $y'(x)$数值；bvp4c 为带边值条件常微分方程积分器的函数命令；odefun 为描述微分方程组的函数；bcfun 为计算边界条件 $g(f(a), f(b), p)=0$ 留数的函数文件；solinit 为一结构，solinit.x 与 solinit.y 分别是初始网格的有序节点与初始估计值，边界值条件分别对应于 a=solinit.x(1)和 b=solinit.x(end)；options 为 bvpset 命令设定的可选函数，可采用系统默认值；$p1$，$p2$⋯为未知参数。

[例 5-7] 求常微分方程 $y''+|y|=0$ 在 $y(0)=2$ 与 $y(4)=-2$ 时的数值解。

[解题过程] 仍使用常用方法改变方程的形式，令 $y_1=y$，$y_2=y_1'$，则 $y_1'=y, y_2'=-|y_1|$；将其写为函数文件 f05071.m；同时写出边界条件留数函数对应的函数文件 f05072.m；最后分别使用结构 solinit 和命令 bvp4c 确定 $x\sim y$之间的关系，并作出 $y\sim x$的关系曲线图。

[算例代码]

```
%例 5-7
solinit =bvpinit([0 1 2 3 4],[1 0]);            %[0 1 2 3 4]为初始网格,[1 0]为初始估计值
sol=bvp4c(@f05071,@twobc,solinit);
    % twoode 与 twobc 分别为微分方程与边界条件留数的函数，solinit 为结构
x=linspace(0,4);                                 %确定 x 范围
y=deval(sol,x);                                  %确定 y 范围
plot(x,y(1,:));                                  %画出 y x 的图形

%定义 f05071 函数（下述代码另存为工作目录下的 f05071.m 文件）
function odefun1= f05071 (x,y);                  %微分方程函数的定义
odefun1=[y(2);-abs(y(1))];

%定义 f05072 函数（下述代码另存为工作目录下的 f05072.m 文件）
function bcfun1= f05072(ya,yb);                  %边界条件留数函数的定义
bcfun1=[ya(1);yb(1)+2];
```

[运行结果]

运行结果如图 5-2 所示。

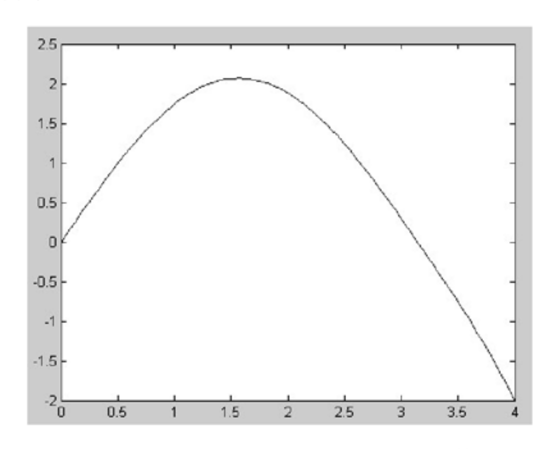


图 5-2　带边值条件微分方程 $y''+|y|=0$ 的数值解

5.5 求解偏微分方程

5.5.1 一般介绍

求解偏微分方程可使用偏微分方程工具箱（Partial Differential Equation Toolbox），它位于安装目录下 Toolbox 子目录的 pde 目录之下。偏微分方程，也称 PDE，是 Partial Differential Equation 的简称，MATLAB 中可通过命令窗口输入命令 pdetool 获得的求解器（如图 5-3 所示）求解偏微分方程。关于 MATLAB 中求解偏微分方程的理论基础和求解方法，可使用其自带的帮助文件学习（MATLAB 7.0→菜单窗口【Help】→【Full Product Family Help】→【Contents】→【Partial Differential Equation Toolbox】，见图 5-4），也可通过其自带的典型算例学习（MATLAB 7.0→菜单窗口【Help】→【Full Product Family Help】→【Demos】→【Toolboxes】→【Partial Differential Equation】→【Command Line Demos】→【Run this demo】，见图 5-5）。

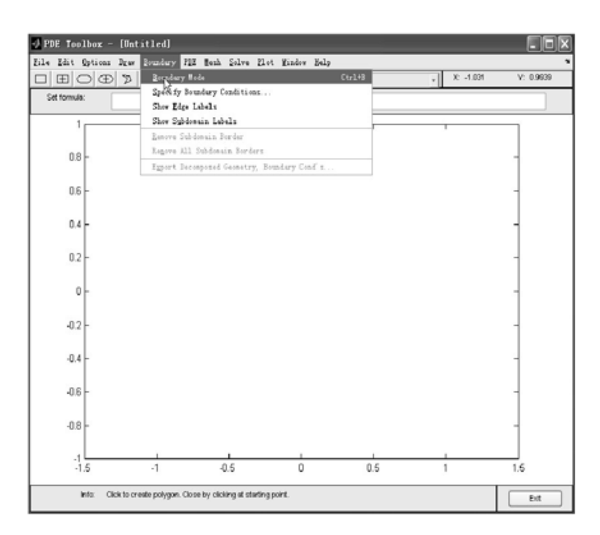


图 5-3 使用 MATLAB 偏微分方程求解器

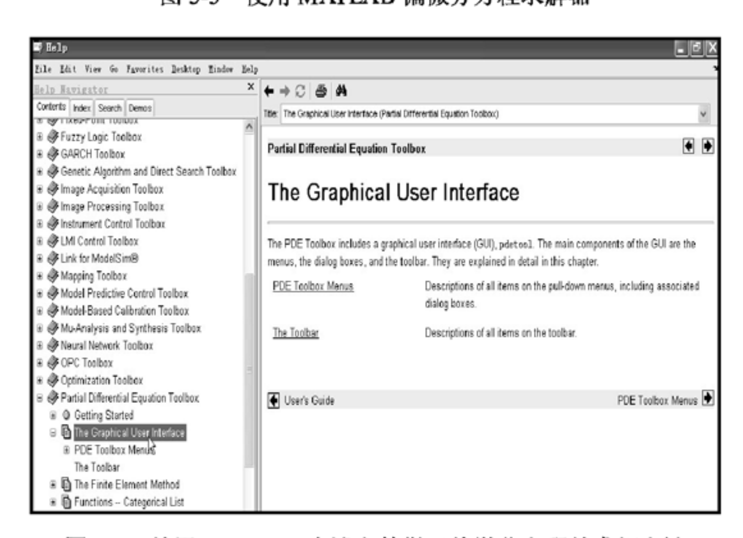
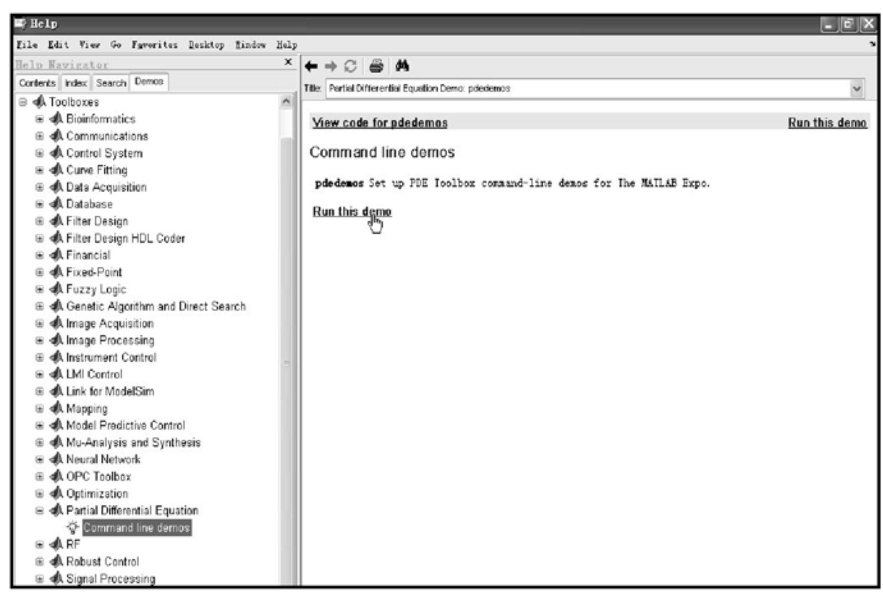
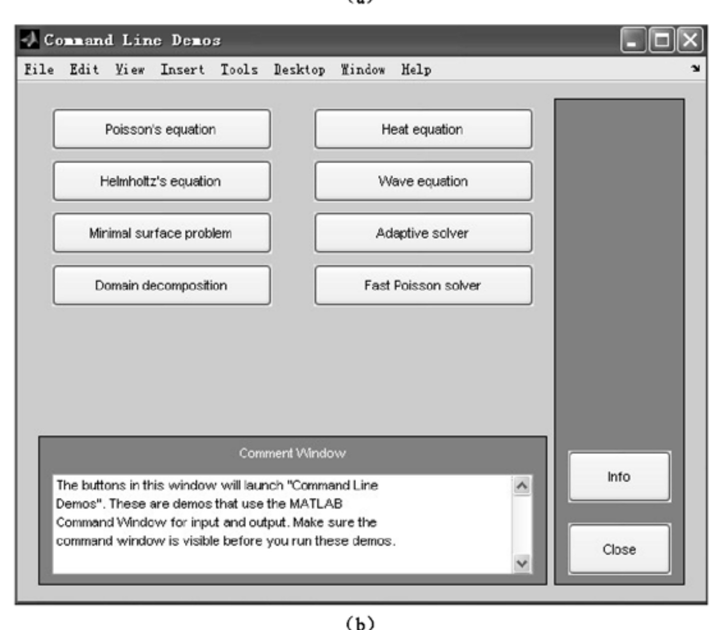


图 5-4 使用 MATLAB 自述文件学习偏微分方程的求解方法



(a)



(b)

图 5-5 使用 MATLAB 自述典型算例学习偏微分方程工具箱

5.5.2 典型算例

下面引用 MATLAB 7.0 自述文件中所附典型算例说明偏微分方程的求解方法，为简便起见，引用时作了一些删减。

[例 5-8] 求解泊松方程 $-\nabla^2 u = 1$ 并与精确解比较，区域为单位圆盘，在圆盘边界上 u=0。

[算例代码]

```
%例 5-8
%问题描述
```

```
g='circleg';                          %单位圆
b='circleb1';                         %边界值为 0
c=1;
a=0;
f=1;
[p,e,t]=initmesh(g,'hmax',1);         %初始网格
error=[];
er=1;
while er > 0.001
    [p,e,t]=refinemesh(g,p,e,t);
    u=assempde(b,p,e,t,c,a,f);
    exact=(1-p(1,:).^2-p(2,:).^2)'/4;
    er=norm(u-exact,'inf');
    error=[error er];
end
pdesurf(p,t,u);                       %求解
pdesurf(p,t,u-exact);                 %误差
```

[运行结果]

运行结果如图 5-6 所示。

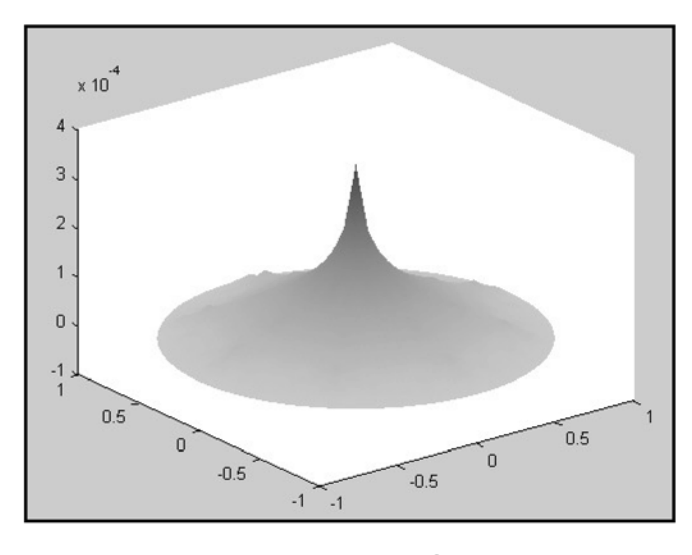


图 5-6　MATLAB 中泊松方程 $-\nabla^2 u=1$ 的求解结果

[例 5-9]　求解热传导方程 $\frac{\partial u}{\partial t}-\nabla^2 u=1$，区域为单位方形，边界上 u=0。

[算例代码]

```
%例 5-9
%问题描述
g='squareg';                          %定义单位方形区域
b='squareb1';                         %边界值为 0
c=1;
a=0;
```

```
f=1;
d=1;
[p,e,t]=initmesh(g);                                    %网格划分
u0=zeros(size(p,2),1);                %定义初始条件：半径 0.4 圆内为 1，其他各处为 0
ix=find(sqrt(p(1,:).^2+p(2,:).^2)<0.4);
u0(ix)=ones(size(ix));
%在时间 0～0.1 之间 20 个点的求解
nframes=20;
tlist=linspace(0,0.1,nframes);
u1=parabolic(u0,tlist,b,p,e,t,c,a,f,d);                 %解决抛物线型方程问题
%为加快画图,在直角网格内进行内插处理
x=linspace(-1,1,31);
y=x;
[unused,tn,a2,a3]=tri2grid(p,t,u0,x,y);
%制作动画
newplot;
Mv = moviein(nframes);
umax=max(max(u1));
umin=min(min(u1));
for j=1:nframes,
    u=tri2grid(p,t,u1(:,j),tn,a2,a3);
    i=find(isnan(u));
    u(i)=zeros(size(i));
    surf(x,y,u);
    caxis([umin umax]);
    colormap(cool);
    axis([-1 1 -1 1 0 1]);
    Mv(:,j) = getframe;
end
%显示动画
movie(Mv,10)
```

[运行结果]

运行结果如图 5-7 所示。

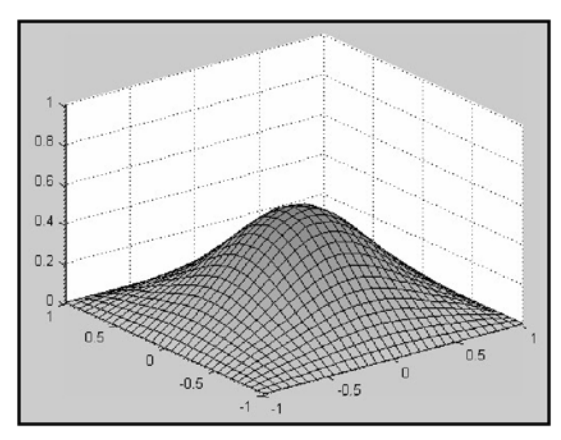


图 5-7 MATLAB 中热传导方程 $\frac{\partial \boldsymbol{u}}{\partial t}-\nabla^2 \boldsymbol{u}=1$ 的求解结果

[例 5-10] 求解 Helmholtz 方程 $-\nabla^2 u - k^2 u = 0$，并研究方形物体反射的波形，波来自于右侧。

[算例代码]

```
%例 5-10
%问题描述
k=60;                                    %入射波波数为 60
g='scatterg';                            %定义求解区域为带方形空洞的圆
b='scatterb';                            %定义入射波的 Dirichlet 条件与外层边界条件
c=1;
a=-k^2;
f=0;
[p,e,t]=initmesh(g);                     %网格初始化
[p,e,t]=refinemesh(g,p,e,t);             %网格修整
[p,e,t]=refinemesh(g,p,e,t);             %网格修整
pdemesh(p,e,t);                          %网格定义为所求偏微分方程的网格
axis equal
%求解振幅
u=assempde(b,p,e,t,c,a,f);
h = newplot;
set(get(h,'Parent'),'Renderer','zbuffer')
pdeplot(p,e,t,'xydata',real(u),'zdata',real(u),'mesh','off');          %绘制波形图
colormap(cool)
%制作反射波动画
m=10;                                                                   %帧数
h = newplot;
hf=get(h,'Parent');
set(hf,'Renderer','zbuffer')
axis tight,
set(gca,'DataAspectRatio',[1 1 1]);
axis off
M=moviein(m,hf);
maxu=max(abs(u));
for j=1:m
    uu=real(exp(-j*2*pi/m*sqrt(-1))*u);
    pdeplot(p,e,t,'xydata',uu,'colorbar','off','mesh','off');
    caxis([-maxu maxu]);
    axis tight;
    set(gca,'DataAspectRatio',[1 1 1]);
    axis off,
    M(:,j)=getframe(hf);
end;
%显示动画
movie(hf,M,50);
```

[运行结果]

运行结果如图 5-8 所示。

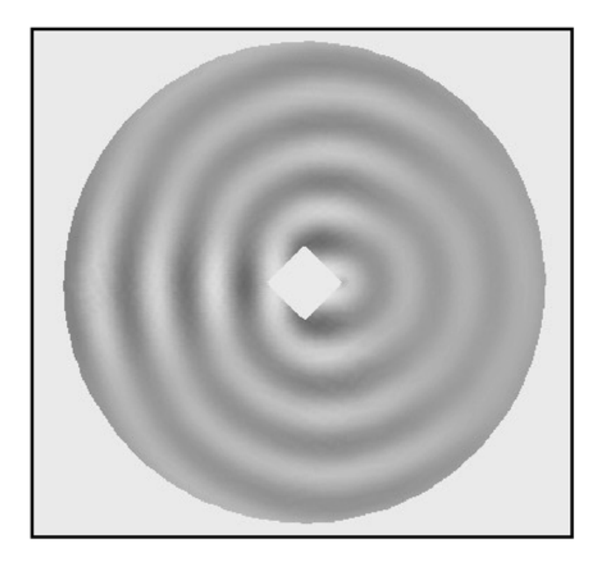


图 5-8 MATLAB 中 Helmholtz 方程 $-\nabla^2 u - k^2 u = 0$ 的求解结果

[例 5-11] 在单位方形区域内求解标准波动方程 $\dfrac{\partial^2 u}{\partial t^2} - \nabla^2 u = 0$。

[算例代码]

```
%例 5-11
%问题描述
g='squareg';                                  %定义单位方形区域
b='squareb3';                                 %定义左右边界值为 0,顶底边界导数值为 0
c=1;
a=0;
f=0;
d=1;
[p,e,t]=initmesh('squareg');                  %划分网格
%初始条件,选择 u(0)=atan(cos(pi/2*x))和 dudt(0)=3*sin(pi*x).*exp(sin(pi/2*y))为边界条件
x=p(1,:)';
y=p(2,:)';
u0=atan(cos(pi/2*x));
ut0=3*sin(pi*x).*exp(sin(pi/2*y));
n=31;
tlist=linspace(0,5,n);                          %在时间段 0~5 内 31 个点上求解
uu=hyperbolic(u0,ut0,tlist,b,p,e,t,c,a,f,d);    %求解该双曲线型偏微分方程
%为加速绘图,在矩形网格上进行内插处理
delta=-1:0.1:1;
[uxy,tn,a2,a3]=tri2grid(p,t,uu(:,1),delta,delta);
gp=[tn;a2;a3];
%制作动画
newplot;
M=moviein(n);
umax=max(max(uu));
umin=min(min(uu));
for i=1:n
```

```
        pdeplot(p,e,t,'xydata',uu(:,i),'zdata',uu(:,i),'zstyle','continuous',...
            'mesh','off','xygrid','on','gridparam',gp,'colorbar','off');
        axis([-1 1 -1 1 umin umax]);
        caxis([umin umax]);
        M(:,i)=getframe;
    end;
    nfps=5;
    movie(M,10,nfps);                                   %显示动画结果
```

[运行结果]

运行结果如图 5-9 所示。

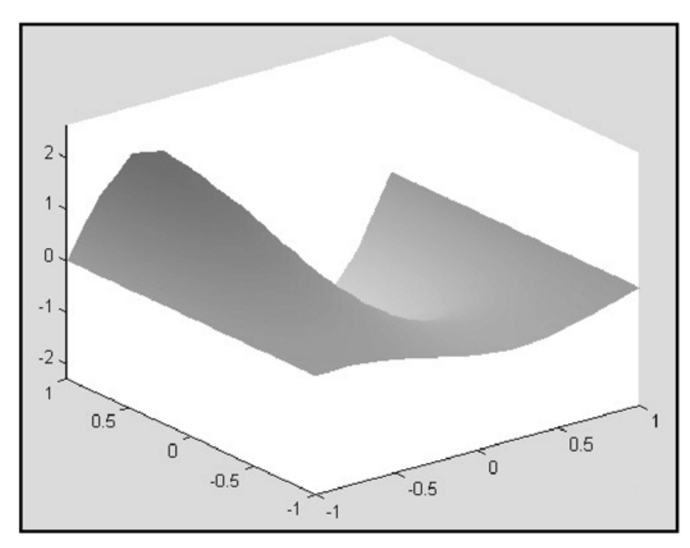


图 5-9　MATLAB 中标准波动方程 $\frac{\partial^2 u}{\partial t^2}-\nabla^2 u=0$ 的求解结果

[例 5-12]　在单位圆内求解最小表面问题 $-\nabla\cdot\left(\frac{1}{\sqrt{1+\nabla|u|^2}\nabla u}\right)=0$，边界条件为 $u=x^2$。

[算例代码]

```
    %例 5-12
    %问题描述
    g='circleg';                                        %定义单位圆
    b='circleb2';                                       %定义边界条件
    c='1./sqrt(1+ux.^2+uy.^2)';
    a=0;
    f=0;
    rtol=1e-3;                                          %定义非线性解的精度
    [p,e,t]=initmesh(g);                                %网格初始化
    [p,e,t]=refinemesh(g,p,e,t);                        %网格修整
    u=pdenonlin(b,p,e,t,c,a,f,'tol',rtol);             %求解最小表面问题
    pdesurf(p,t,u);                                     %显示求解结果
```

[运行结果]

运行结果如图 5-10 所示。

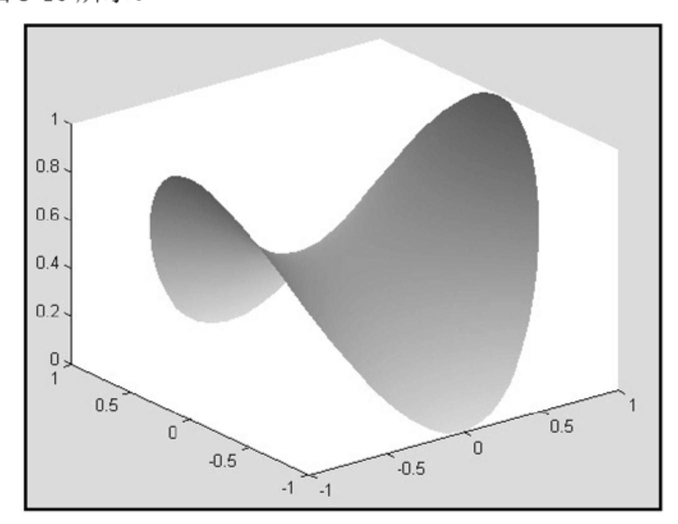


图 5-10　MATLAB 中最小表面问题 $-\nabla\cdot\left(\frac{1}{\sqrt{1+\nabla|u|^2}\nabla u}\right)=0$ 的求解结果

[例 5-13]　求解泊松方程 $-\nabla^2 u=\delta(x,y)$ 并与精确解 $u=-\frac{1}{2\pi}\ln r$ 比较，求解域为单位圆，边界上 $u=0$。

[算例代码]

```
%例 5-13
%问题描述
g='circleg';                                    %定义单位圆
b='circleb1';                                   %定义 0 边界
c=1;
a=0;
f='circlef';                                    %定义点源
%使用内部函数 tripick 进行网格划分
[u,p,e,t]=adaptmesh(g,b,c,a,f,'tripick','circlepick','maxt',2000,'par',1e-3);
pdemesh(p,e,t);
axis equal                                      %等轴
pdeplot(p,e,t,'xydata',u,'zdata',u,'mesh','off');   %求解结果
%与精确解比较
x=p(1,:)';
y=p(2,:)';
r=sqrt(x.^2+y.^2);
uu=-log(r)/2/pi;
pdeplot(p,e,t,'xydata',u-uu,'zdata',u-uu,'mesh','off');
```

[运行结果]

运行结果如图 5-11 所示。

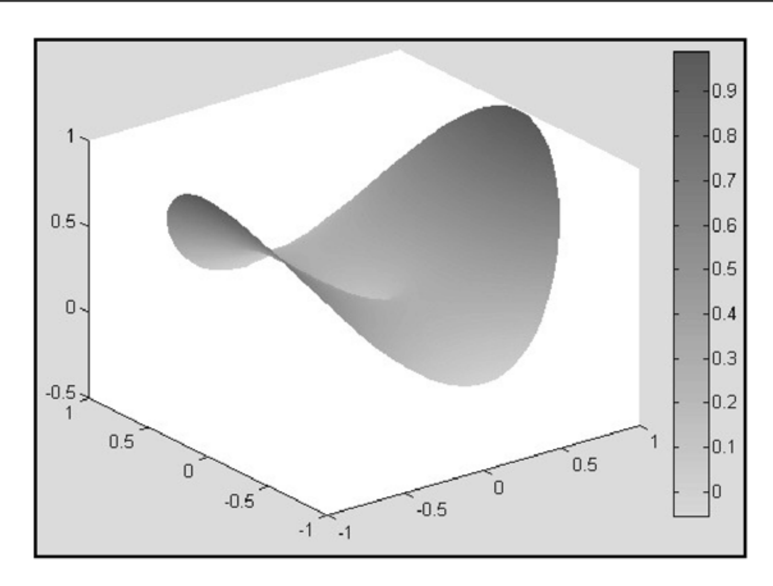


图 5-11　MATLAB 中泊松方程 $-\nabla^2 u = \delta(x, y)$ 的求解结果

[例 5-14]　使用子区域分解法求解泊松方程 $-\nabla^2 u = 1$，求解域为 L 形薄膜，边界上 $u = 0$。

[算例代码]

```
%例 5-14
%问题描述
g='lshapeg';                                    %定义 L 形薄膜
b='lshapeb';                                    %定义 0 边界
c=1;
a=0;
f=1;
time=[];                                        %给函数 ASSEMPDE 输入时间变量
[p,e,t]=initmesh(g);                            %网格初始化
[p,e,t]=refinemesh(g,p,e,t);                    %网格修整
[p,e,t]=refinemesh(g,p,e,t);
np=size(p,2);
cp=pdesdp(p,e,t);                               %寻找公共点
nc=length(cp);                                  %分配空间
C=zeros(nc,nc);                                 % Schur 余数
FC=zeros(nc,1);
%集合范围 1 并更新余数
[i1,c1]=pdesdp(p,e,t,1);ic1=pdesubix(cp,c1);
[K,F]=assempde(b,p,e,t,c,a,f,time,1);
K1=K(i1,i1);d=symamd(K1);i1=i1(d);
K1=chol(K1(d,d));B1=K(c1,i1);a1=B1/K1;
C(ic1,ic1)=C(ic1,ic1)+K(c1,c1)-a1*a1';
f1=F(i1);e1=K1'\f1;FC(ic1)=FC(ic1)+F(c1)-a1*e1;
%集合范围 2 并更新余数
[i2,c2]=pdesdp(p,e,t,2);ic2=pdesubix(cp,c2);
[K,F]=assempde(b,p,e,t,c,a,f,time,2);
K2=K(i2,i2);d=symamd(K2);i2=i2(d);
K2=chol(K2(d,d));B2=K(c2,i2);a2=B2/K2;
```

```
C(ic2,ic2)=C(ic2,ic2)+K(c2,c2)-a2*a2';
f2=F(i2);e2=K2'\f2;FC(ic2)=FC(ic2)+F(c2)-a2*e2;
%集合范围 3 并更新余数
[i3,c3]=pdesdp(p,e,t,3);ic3=pdesubix(cp,c3);
[K,F]=assempde(b,p,e,t,c,a,f,time,3);
K3=K(i3,i3);d=symamd(K3);i3=i3(d);
K3=chol(K3(d,d));B3=K(c3,i3);a3=B3/K3;
C(ic3,ic3)=C(ic3,ic3)+K(c3,c3)-a3*a3';
f3=F(i3);e3=K3'\f3;FC(ic3)=FC(ic3)+F(c3)-a3*e3;
%求解
u=zeros(np,1);
u(cp)=C\FC;                                  %求出公共点
u(i1)=K1\(e1-a1'*u(c1));                     % SD1 内点
u(i2)=K2\(e2-a2'*u(c2));                     % SD2 内点
u(i3)=K3\(e3-a3'*u(c3));                     % SD3 内点
%绘图
pdesurf(p,t,u)
```

[运行结果]

运行结果如图 5-12 所示。

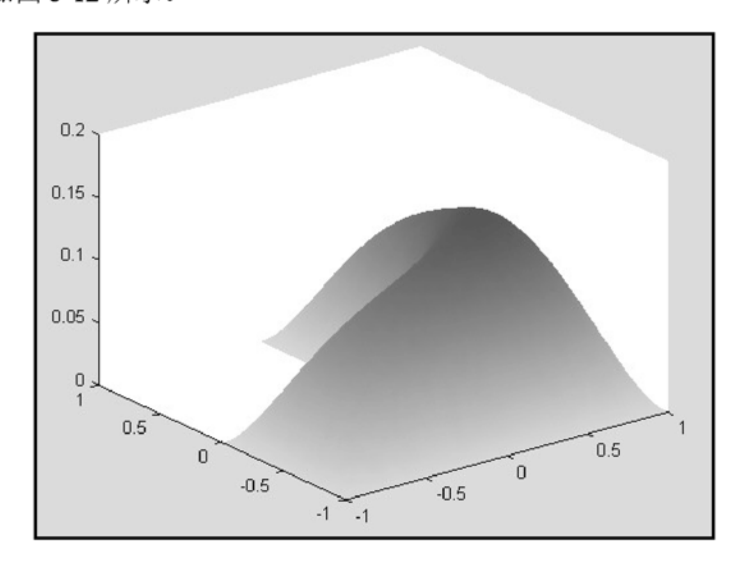


图 5-12 MATLAB 中使用子区域分解法对泊松方程 $-\nabla^2 u = 1$ 的求解结果

[例 5-15] 求解泊松方程 $-\nabla^2 U = 3x^2$，并与标准 PDE 求解器求解结果比较，求解域为带有 Dirichlet 边界条件的单位方形区域。

[算例代码]

```
%例 5-15
%问题描述
g='squareg';                          %定义方形区域
b='squareb4';                         %三面边界值为 0，右端边界值呈半正弦分布
c=1;
a=0;
```

```
f='3*x.^2';
%使用 17*17 节点的规则网格
n=16;
[p,e,t]=poimesh(g,n);
pdemesh(p,e,t);
axis equal
%快速求解
tm=cputime;
u=poisolv(b,p,e,t,f);
cputime-tm
%与非快速求解结果比较
tm=cputime;
u1=assempde(b,p,e,t,c,a,f);
cputime-tm
%显示求解结果
pdesurf(p,t,u)
%使用 65*65 节点的网格
n=64;
[p,e,t]=poimesh(g,n);
%快速求解
tm=cputime;
u=poisolv(b,p,e,t,f);
cputime-tm
%与非快速求解结果比较
tm=cputime;
u1=assempde(b,p,e,t,c,a,f);
cputime-tm
```

[运行结果]

运行结果如图 5-13 所示。

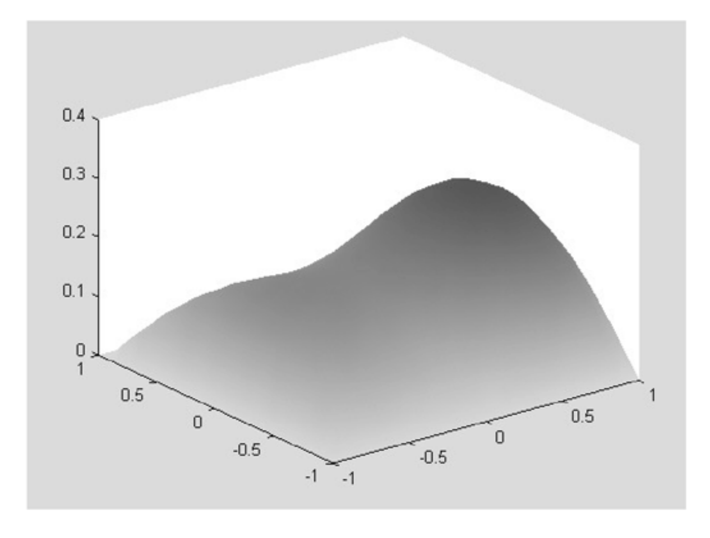


图 5-13 MATLAB 中泊松方程 $-\nabla^2 U = 3x^2$ 的求解结果

习题 5

1．试用 MATLAB 计算函数 $f(x)=\sin(x)+\cos(x)$在[−1,1]中的零点。

2．试用 MATLAB 求函数 $f(x)=x^2-2x+2$ 在区间(0,2)上的极值

3．试用 MATLAB 求函数 $f(x,y,z)=x^2+\sin y-z^2y^2$ 的极值，初值为 $x_0=-0.3$，$y_0=-1.1$，$z_0=0.15$。

4．试用 MATLAB 计算函数 $f(x)=\dfrac{\sin x}{x}$在区间[1,2]中的零点与极值。

5．试用 MATLAB 求下列常微分方程的数值解。

（1） $y''-(1-y^2)y'+y=0$，$y(1)=2$，$y(2)=0$，$x\in[0,20]$

（2） $y''-y'+y=0$，$y(1)=2$，$y(2)=0$，$x\in[0,20]$

6．试用 MATLAB 求解泊松方程 $-\nabla^2u=1$，区域为单位圆盘，在圆盘边界上 $u=0$。

7．试用 MATLAB 求解热传导方程 $\dfrac{\partial u}{\partial t}-\nabla^2u=1$，区域为单位正方形，边界上 $u=0$。

8．试用 MATLAB 求解方程 $-\nabla^2u-k^2u=0$。

9．试用 MATLAB 在单位方形区域内求解波动方程 $\dfrac{\partial^2u}{\partial t^2}-\nabla^2u=0$。

第 6 章　符 号 计 算

本章要点：

☑ MATLAB 中的符号计算概述；
☑ MATLAB 中的符号定义方法与符号函数；
☑ MATLAB 中符号的初等代数运算、复合函数与反函数、极限、泰勒展开、级数求和、微分、积分、线性代数运算、线性方程与微分方程、数学变换求解的算法实现。

6.1　概述

MATLAB 7.0 中的命令，如果其操作对象不是数值而是符号（符号表达式或符号数组），相应的计算称为符号计算，例如，当 a，b 是变量时，计算积分 $\int_a^b x\mathrm{d}x$ 就是符号计算。在符号计算中，通过符号常数、符号变量及有关符号操作形成符号表达式。

由于 MATLAB 7.0 中符号计算函数是数值计算函数的重载，符号计算工具箱采用的函数和数值计算的函数有一部分同名，为了得到准确的在线帮助，不能直接使用“help 函数名”的格式、而应该使用“help sym/函数名”的格式。

[例 6-1]　查询逆矩阵符号计算。

```
%例 6-1
help sym/inv
```

[运行结果]

```
INV     Symbolic matrix inverse.
    INV(A) computes the symbolic inverse of A
    INV(VPA(A)) uses variable precision arithmetic.

    Examples:
       Suppose B is
          [ 1/(2-t), 1/(3-t) ]
          [ 1/(3-t), 1/(4-t) ]

       Then inv(B) is
          [        -(-3+t)^2*(-2+t), (-3+t)*(-2+t)*(-4+t) ]
          [ (-3+t)*(-2+t)*(-4+t),        -(-3+t)^2*(-4+t) ]

       digits(10);
```

```
    inv(vpa(sym(hilb(3))));

See also vpa.

Reference page in Help browser
    doc sym/inv
```

运行 MATLAB 7.0→菜单窗口【Help】→【Full Product Family Help】→【Demos】→【Toolboxes】→【Symbolic Math】→【Command Line Demos】→【Run this demo】，可以浏览符号计算的简单介绍和有关示例，如图 6-1 所示。

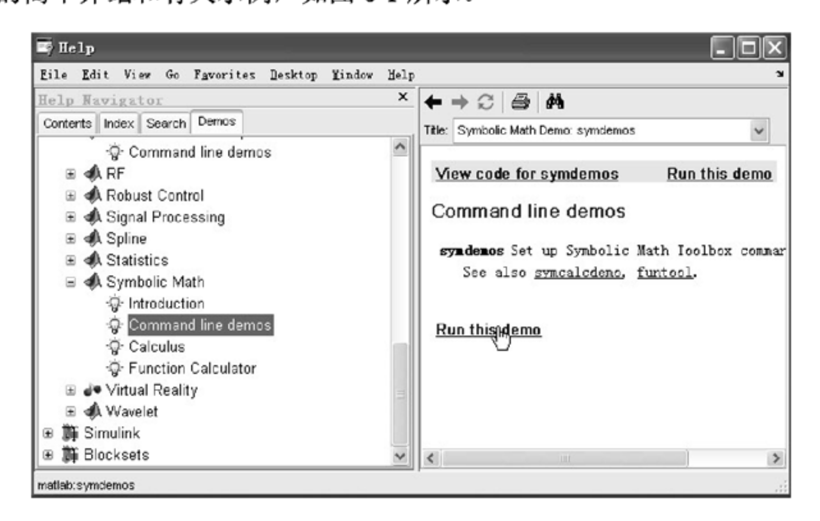


图 6-1 使用 MATLAB 自述文件浏览符号计算

6.2 符号定义

6.2.1 基本符号的定义

[调用命令]

sym，syms

[调用格式]

f=sym('a', 'b', 'c');

syms a b c

重要提示：syms a b c 更为简洁一些，它与“sym a; sym b; sym c;”或“syms a; syms b; syms c;”等价、但不能写为“sym a b c”（涉及多个符号时只能用命令 syms 而不能用命令 sym），命令“syms a b c”中没有逗号，句末分号可以有、也可以没有。

符号计算的结果是符号或符号表达式，如果其中的符号要用具体数值代替，可以使用命令 subs，将 A2 中的符号 a 以数值 1 代替可以使用 A3=subs(A2,a,1)。要注意的是，使用命令 subs 时不能同时替换多个变量、一次只能替换一个变量，见例 6-2。

[例 6-2] 计算行列式 $\boldsymbol{A}=\begin{vmatrix} a & b \\ c & d \end{vmatrix}$，当 a=1、b=2、c=3、d=4 时计算 $\boldsymbol{A}$ 的数值结果。

[算例代码]

```
%例 6-2
clear;
syms a b c d;
A1=[a,b;c,d];
A2=det(A1);
A3=subs(A2,a,1);
A3=subs(A3,b,2);
A3=subs(A3,c,3);
A3=subs(A3,d,4)
```

[运行结果]

```
A3 = –2
```

6.2.2 符号函数

生成符号函数方法有两个，即使用符号表达式和 m 文件。

1．使用符号表达式

在符号定义以后，可以使用符号表达式直接形成符号函数。

[例 6-3] 使用符号表达式分别形成符号函数 $f=\sin(x)\cos(y)$、$g=\ln(x^2+y^2)$。

[算例代码]

```
%例 6-3
clear;
syms x y;
f=sin(x)*cos(y)
g=log(x^2+y^2)
```

[运行结果]

```
f =sin(x)*cos(y)
g =log(x^2+y^2)
```

2．使用 m 文件

实际编程中，使用编制 m 文件形成符号函数然后再行调用，有时候更为方便一些。

[例 6-4] 编制一个符号函数文件，将矩阵的主对角线元素置换为符号 t。

[算例代码]

```
%例 6-4
%定义 f0604 函数（下述代码另存为工作目录下的 f0604.m 文件）
function A= f0604(N)
%这是一个符号函数。
%它将矩阵的主对角线元素变为符号 t
syms t;
for i=1:N
    for j=1:N
        A(i,j)=t;
    end
end
```

6.3 符号运算

6.3.1 初等代数运算

1．符号的加减乘除运算

符号的加、减、乘、除运算与数值计算中的相应运算符相同，即分别使用+、–、*、/。要注意的是，进行符号关系运算时，不能使用“大于”、“小于”、“大于等于”和“小于等于”等命令进行比较，但可以使用是否等于的判断，即两个符号表达式相等返回 1，否则返回 0。

2．符号表达式化简

符号表达式化简的有关函数见表 6-1。

表 6-1 MATLAB 中的符号表达式化简函数

函数名	意义	调用格式
collect	合并同类项，即对相同次幂的项进行合并	collect(S,v)或 collect(S)
factor	因式分解，得到有理系数多项式	factor(S)
numden	使分子、分母分离	[N,D]=numden(S) (N 和 D 分别为分子和分母)
simplify	对符号矩阵化简	simplify(S)
expand	展开多项式或函数	expand(S)
horner	生成嵌套多项式	horner(S)
simple	化简为最简形式	simple(S)
subexpr	将重复出现的字符串用变量代替，重新生成表达式	subexpr(S,char)
subs	将重复出现的字符用变量代替，重新生成表达式	subs(S,char,value),见例 4-2

[例 6-5] 合并$(x+y)(e^x+y)$中的同类项、对x^2-y^2进行因式分解、展开多项式$\cos(x+y)$、对$\cos^2 x+\sin^2 x$进行化简。

[算例代码]

```
%例 6-5
clear                                   %清除内存变量
syms x y;                               %定义符号
R1=collect(exp((x)+x)*(x+y))            %合并同类项
R2=factor(x^2-y^2)                      %因式分解
R3=expand(cos(x+y))                     %多项式展开
R4=simple(cos(x)^2+sin(x)^2)            %将多项式化为最简式
```

[运行结果]

```
R1 =exp(2*x)*x+exp(2*x)*y
R2 =(x–y)*(x+y)
R3 =cos(x)*cos(y)–sin(x)*sin(y)
R4 =1
```

6.3.2 复合函数

[函数命令]

compose

[调用格式]

h= compose(f,g)

h 为结果表达式，为函数 $f= f(x)$和 $g= g(y)$的复合函数 $f(g(y))$。

[其他格式]

h= compose(f,g,z)，返回函数 $f= f(x)$和 $g= g(z)$的复合函数 $f(g(z))$。

h= compose(f,g,x,z)，返回函数 $f= f(x)$和 $g= g(z)$的复合函数 $f(g(z))$，x是函数 f的自变量。

h= compose(f,g,x,z)，返回函数 $f= f(x)$和 $g= g(z)$的复合函数 $f(g(z))$，x和 z分别是函数 f 和 g的自变量。

[例 6-6] 已知 $f=\dfrac{1}{1+x^2}$，$g= \sin(y)$，求复合函数 $f(g(y))$。

[算例代码]

```
%例 6-6
clear
syms x y;
f = 1/(1 + x^2);                                    %定义函数 f
g = sin(y);                                         %定义函数 f
h= compose(f,g)                                     %求复合函数 f(g(y))
```

[运行结果]

h =1/(1+sin(y)^2)

6.3.3 反函数

[函数命令]

finverse

[调用格式]

h= finverse(f)

h 为函数 f的反函数。

[其他格式]

g=finverse(f,v)，返回函数 f关于变量 v的反函数。

[例 6-7] 求函数 $f=\dfrac{1}{\text{tg}(x)}$ 的反函数。

[算例代码]

```
%例 6-7
clear
syms x;
f = 1/tan(x);                                       %定义函数 f
g= finverse (f)                                     %求 f 的反函数
```

[运行结果]

g =atan(1/x)

6.3.4 求极限

[函数命令]

limit

[调用格式]

g= limit(f,x,a,'left')

*g*为结果表达式，是函数 *f*对 *x*在 *x*=*a* 处的左极限（'left'），函数 *f*可以是一个函数或一个矩阵；求右极限则将 'left' 改为 'right'；而求无穷大处的极限，则将 a,'left' 改为 inf。

[例 6-8] 分别求函数 $\boldsymbol{X}=a\left(1+\frac{1}{x}\right)^x$ 与矩阵 $\boldsymbol{Y}=\begin{vmatrix} a\left(1+\frac{1}{x}\right)^x & \mathrm{e}^{-x} \\ \sin x/x & 1/x \end{vmatrix}$ 在 *x* 是无穷大处的左极限，这里 *a* 为系数。

[算例代码]

```
%例 6-8
clear
syms x a;
R11=a*(1+1/x)^x;                              %目的函数 R11
R12=[a*(1+1/x)^x,exp(-x),sin(x)/x,1/x];       %目的矩阵 R12
R21=limit(R11,x,inf, 'left')                  %求 R11 对 x 在正无穷处的左极限
R22=limit(R12,x,inf, 'left')                  %求 R12 对 x 在正无穷处的左极限
```

[运行结果]

R21 =exp(1)*a

R22 =[exp(1)*a,   0,   0,   0]

6.3.5 泰勒展开

[函数命令]

taylor

[调用格式]

g= taylor(f,x,a,k)

*g*为结果表达式，将函数 *f*对 *x*在 *x*=*a* 处泰勒（Taylor）展开到第（*k*–1）项。

重要提示：命令中的 *k*表示展开到第（*k*–1）项而不是第 *k*项（根据数学中的泰勒展开公式很容易理解这一点）；由于展开式仍为关于 *x*的表达式（这似乎是 MATLAB 7.0 的一个漏洞），可以使用命令 subs 求得准确的结果。

[例 6-9] 将函数 $f=\sin x$ 在 $x=\pi/2$ 处展开到第 5 项。

[算例代码]

```
%例 6-9
clear
syms x;
```

```
R1=taylor(sin(x),pi/2,6)
R11=subs(R1,x, pi/2)
```

[运行结果]

```
R1=1-1/2*(x-1/2*pi)^2+1/24*(x-1/2*pi)^4
R11=1
```

6.3.6 级数求和

[函数命令]

symsum

[调用格式]

g= symsum(s,v,a,b)

g为对表达式 s 的符号变量 v 从 v=a 到 v=b 进行求和的结果；如果 v 为系统默认的符号变量 k，v 这一项可省略；如果 v，a 这两项省略，表示符号变量 k 从 k=0 到 k=b 进行求和；如果 v，a，b 这三项省略，表示符号变量 k 从 0 到 k-1 项进行求和。

重要提示：如果结果 g 不是一个表达式，结果虽然存在（收敛），比如有关热传导问题的解，则 MATLAB 并不给出解答，此时可通过问题的特点、用循环语句等命令解决问题。

[例 6-10] 计算 $R1=\sum_{k=0}^{k=\infty}\frac{x^k}{k!}$。

[算例代码]

```
%例 6-10
clear
syms k x y a b;
R1=symsum(x^k/sym('k!'),k,0,inf)                    % 对 x^k/k!中的 k 由 0 到无穷大求和
```

[运行结果]

```
R11 = exp(x)
```

[例 6-11] 地下某一位置处饱和土单元体的固阶微分方程 $C_v\frac{\partial^2 u}{\partial z^2}=-\frac{\partial u}{\partial t}$ 在单面排水条件下的解为 $U_z=1-\frac{8}{\pi^2}\sum_{k=0}^{k=\infty}\frac{1}{(2k+1)^2}e^{-(2k+1)^2\pi^2T_v/4}$，$C_v$为土的竖向固结系数，$u$是超孔隙水压力，$z$是计算点深度，$t$是时间，$U_z$是平均固结度，$k$为非负整数，$T_v$是时间因数（$T_v$=$C_v t/H^2$），$H$ 为土层厚度，试在半对数坐标系中作出 U_z与 T_v的关系曲线。

[算例代码]

```
%例 6-11
clear
for i=1:1001
    T=0;
    Tv(i)=(i-1)/100;
    for k=0:1:5
        T=T+8/pi/pi/(2*k+1)/(2*k+1)*exp(-(2*k+1)*(2*k+1)*pi*pi*Tv(i)/4);
    end
```

```
    Uz(i)=1-T;
end
semilogx(Tv,Uz)                                 %取半对数坐标系(X 轴取对数)
set(gca,'YLim',[0,1],'YDir','reverse')          %确定 Y 轴的数据范围与坐标轴方向
grid on;                                        %增加网格线
xlabel('时间因数 Tv');ylabel('平均固结度 Uz');      %增加坐标轴标签
```

[运行结果]

作图有关命令见本书“图形处理”部分，运行结果如图 6-2 所示。

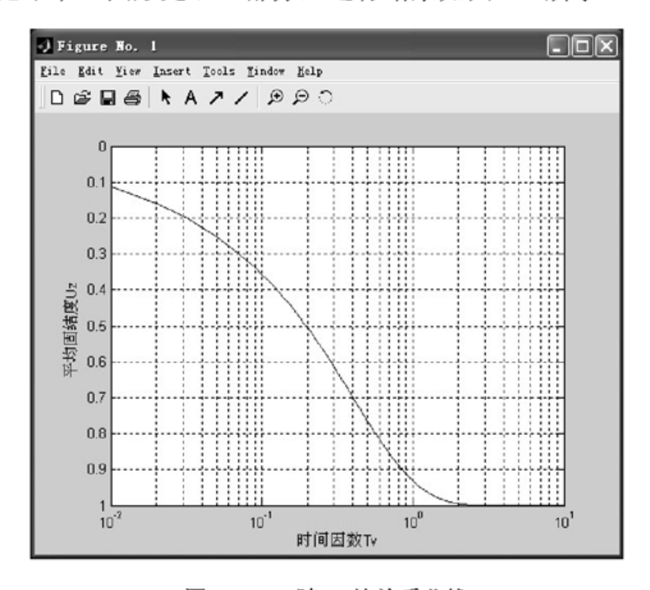


图 6-2 U_z随 T_v的关系曲线

6.3.7 符号微分

1．基本方法

[函数命令]

diff

[调用格式]

g= diff(f,x,k)

g 为求函数 *f* 对 *x* 的 *k* 阶微分的结果表达式，*f* 可以是单一函数、也可以是一个函数矩阵；如果 *x* 一项省略，则表示默认自变量是 *x*；如果 *k* 一项省略，则表示是求一阶微分。

[例 6-12] 分别求函数矩阵 $X = a\left(1+\frac{1}{x}\right)^x$ 与矩阵 $Y = \begin{vmatrix} a\left(1+\frac{1}{x}\right)^x & e^{-x} \\ \sin x/x & 1/x \end{vmatrix}$ 对 *x* 的三阶微分，*a* 为常数。

[算例代码]

```
%例 6-12
clear
syms x a;
```

```
R11=a*(1+1/x)^x;                             %目的函数 R11
R12=[a*(1+1/x)^x,exp(-x),sin(x)/x,1/x];      %目的矩阵 R12
R21=diff(R11,x,3)                            %求 R11 对 x 的三阶微分
R22= diff(R12,x,3)                           %求 R12 对 x 的三阶微分
```

[运行结果]

```
R21 =
    a*(1+1/x)^x*(log(1+1/x)-1/x/(1+1/x))^3-3*a*(1+1/x)^x*(log(1+1/x)-1/x/(1+1/x))/x^3|(1+1/x)^2+
    3*a*(1+1/x)^x/ x^4/(1+1/x)^2-2*a*(1+1/x)^x/x^5/(1+1/x)^3
R22 =
    [a*(1+1/x)^x*(log(1+1/x)-1/x/(1+1/x))^3-3*a*(1+1/x)^x*(log(1+1/x)-1/x/(1+1/x))/x^3/(1+1/x)^
    2+3*a*(1+1/x)^x/x^4/(1+1/x)^2-2*a*(1+1/x)^x/x^5/(1+1/x)^3,
    -exp(-x),
    -cos(x)/x+3*sin(x)/x^2+6*cos(x)/x^3-6*sin(x)/x^4,
    -6/x^4]
```

2．Jacobi矩阵

[函数命令]

jacobian

[调用格式]

g= jacobian([F,G],[x,y])

g为函数 F，G对 x，y的 Jacobi 矩阵$\dfrac{\partial(F,G)}{\partial(x,y)}$的结果表达式。

[其他格式]

g=jacobian([F,G,H],[x,y,z])为函数 F，G，H对 x，y，z的 Jacobi 矩阵$\dfrac{\partial(F,G,H)}{\partial(x,y,z)}$。

[例 6-13] 已知 $F=az\cos(x)\sin(y)$，$G=bz\cos(xy)$，$H=cz\sin(x)$，求 Jacobi 矩阵$\dfrac{\partial(F,G,H)}{\partial(x,y,z)}$，这里 a，b，c为系数。

[算例代码]

```
%例 6-13
clear
syms x y z a b c;
F=a*z*cos(x)*sin(y);                 %函数 F
G=b*z*cos(x*y);                      %函数 G
H=c*z*sin(x);                        %函数 H
R=jacobian([F,G,H],[x,y,z])          %计算 Jacobi 矩阵
```

[运行结果]

```
R=
[ -a*z*sin(x)*sin(y),   a*z*cos(x)*cos(y),      a*cos(x)*sin(y)]
[    -b*z*sin(x*y)*y,      -b*z*sin(x*y)*x,          b*cos(x*y)]
[          c*z*cos(x),                    0,            c*sin(x)]
```

6.3.8 符号积分

[函数命令]

int

[调用格式]

g1= int(f,x), g2= int(f,x,a,b)

$g1$ 和 $g2$ 为结果表达式，$g1$ 为函数 f对 x的不定积分，$g2$ 为函数 f对 x在区间$[a,b]$上的定积分。

重要提示：由符号定积分的求法可以方便地应用于二重积分、三重积分的数值计算。

[例 6-14] 计算积分 $R1=\int\frac{x}{1+z^2}\mathrm{d}z$、$R2=\int_0^1 x\ln(1+x)\mathrm{d}z$与 $R3=\int_x^{x+1}\int_0^1(x^2+y^2+1)\mathrm{d}x\mathrm{d}y$。

[算例代码]

```
%例 6-14
clear;
syms x y z;
R1=int(x/(1+z^2),z)                          %求 x/(1+z^2)关于 z 的不定积分
R2=int(x*log(1+x),0,1)                       %求 x*log(1+x)在[0,1]内的积分值
R3=int(int(x^2+y^2+1,y,x,x+1),x,0,1)         %求 x^2+y^2+1 在 y=[x,x+1]，x=[0,1]之中的积分值
```

[运行结果]

```
R1 =x*atan(z)
R2 =1/4
R3 =5/2
```

6.3.9 线性代数运算

[函数命令]

符号线性代数运算使用的函数命令，与数值及其表达式的命令基本一致，表 6-2 为 MATLAB 7.0 中符号线性代数运算常用的函数命令，下面用一综合实例作简略说明。

表 6-2 符号线性代数运算一览表*

函数名	数学意义	调用格式
det	求行列式的值	det(A)
null	生成一个空矩阵	null(A)
colspace	生成基本列空间	colspace(A)
diag	求矩阵的对角元素	diag(A)
rank	求矩阵的秩	rank(A)
expm	求矩阵的幂	expm(A)
inv	求矩阵的逆	inv(A)
rref	生成类似上三角阵	rref(A)
triu	生成上三角阵	triu(A)
tril	生成下三角阵	tril(A)
Poly	求特征多项式	poly(A)

续表

函 数 名	数 学 意 义	调 用 格 式
eig	求特征值与特征向量	[V,D]=eig(A)
svd	矩阵的奇异值分解	svd(A)
jordan	求矩阵的 Jordan 标准型	jordan(A)

* 说明：表中 ***A*** 为方阵，***V*** 和 ***D*** 分别为特征向量满阵与特征值对角矩阵(***A*V= V*D***)。

[例 6-15] 举例说明 MATLAB 7.0 中符号线性代数的运算。

[算例代码]

```
%例 6-15
clear;
syms x y z a b;
A=[a b;x y];                                          %定义符号矩阵
R01= det (A)
R02= null(A)
R03= colspace(A)
R04= diag(A)
R05= rank(A)
R06= expm(A)
R07= inv(A)
R08= rref(A)
R09= triu(A)
R10= tril(A)
R11= poly(A)
[R12,R13]= eig(A)
R14= svd(A)
R15= jordan(A)
```

[运行结果]

```
R01 =a*y-b*x
R02 =[ empty sym ]
R03 =
  [ 0, 1]
  [ 1, 0]
R04 =
  [ a]
  [ y]
R05 =2
R06 =
   [-1/2*(-(a^2-2*a*y+y^2+4*b*x)^(1/2)*exp(1/2*a+1/2*y+1/2*(a^2-2*a*y+y^2+4*b*x)^(1/2))+
   a*exp(1/2*a+1/2*y-1/2*(a^2-2*a*y+y^2+4*b*x)^(1/2))-a*exp(1/2*a+1/2*y+1/2*(a^2-2*a*y+
   y^2+4*b*x)^(1/2))-y*exp(1/2*a+1/2*y-1/2*(a^2-2*a*y+y^2+4*b*x)^(1/2))+y*exp(1/2*a+1/2*y+
   1/2*(a^2-2*a*y+y^2+4*b*x)^(1/2))-(a^2-2*a*y+y^2+4*b*x)^(1/2)*exp(1/2*a+1/2*y-1/2*(a^2-
   2*a*y+y^2+4*b*x)^(1/2)))/(a^2-2*a*y+y^2+4*b*x)^(1/2),
```

```
    b*(-exp(1/2*a+1/2*y-1/2*(a^2-2*a*y+y^2+4*b*x)^(1/2))+exp(1/2*a+1/2*y+1/2*(a^2-2*a*y+
    y^2+4*b*x)^(1/2)))/(a^2-2*a*y+y^2+4*b*x)^(1/2)]
    [x*(-exp(1/2*a+1/2*y-1/2*(a^2-2*a*y+y^2+4*b*x)^(1/2))+exp(1/2*a+1/2*y+1/2*(a^2-2*a*y+
    y^2+4*b*x)^(1/2)))/(a^2-2*a*y+y^2+4*b*x)^(1/2),
    1/2*((a^2-2*a*y+y^2+4*b*x)^(1/2)*exp(1/2*a+1/2*y+1/2*(a^2-2*a*y+y^2+4*b*x)^(1/2))-
    y*exp(1/2*a+1/2*y-1/2*(a^2-2*a*y+y^2+4*b*x)^(1/2))+y*exp(1/2*a+1/2*y+1/2*(a^2-2*a*y+y^2+
    4*b*x)^(1/2))+a*exp(1/2*a+1/2*y-1/2*(a^2-2*a*y+y^2+4*b*x)^(1/2))-a*exp(1/2*a+1/2*y+
    1/2*(a^2-2*a*y+y^2+4*b*x)^(1/2))+(a^2-2*a*y+y^2+4*b*x)^(1/2)*exp(1/2*a+1/2*y-1/2*(a^2-
    2*a*y+y^2+4*b*x)^(1/2)))/(a^2-2*a*y+y^2+4* b*x)^(1/2)]
R07 =
    [ -y/(-a*y+b*x),    b/(-a*y+b*x)]
    [    x/(-a*y+b*x), -a/(-a*y+b*x)]
R08 =
    [ 1, 0]
    [ 0, 1]
R09 =
    [ a, b]
    [ 0, y]
R10 =
    [ a, 0]
    [ x, y]
R11 =
    t^2-t*y-a*t+a*y-b*x
R12 =
    [-(-1/2*a+1/2*y-1/2*(a^2-2*a*y+y^2+4*b*x)^(1/2))/x,-(-1/2*a+1/2*y+1/2*(a^2-2*a*y+y^2+
    4*b*x)^(1/2))/x]
    [1,1]
R13 =
    [1/2*a+1/2*y+1/2*(a^2-2*a*y+y^2+4*b*x)^(1/2),0]
    [0, 1/2*a+1/2*y-1/2*(a^2-2*a*y+y^2+4*b*x)^(1/2)]
R14 =
    [1/2*(2*a*conj(a)+2*y*conj(y)+2*b*conj(b)+2*x*conj(x)+2*(a^2*conj(a)^2-2*a*conj(a)*y*conj(y)+
    2*a*conj(a)*b*conj(b)+2*a*conj(a)*x*conj(x)+y^2*conj(y)^2+2*b*conj(b)*y*conj(y)+
    2*y*conj(y)*x*conj(x)+b^2*conj(b)^2-2*b*conj(b)*x*conj(x)+x^2*conj(x)^2+4*a*conj(x)*y*
    conj(b)+4*b*conj(y)*x*conj(a))^(1/2))^(1/2)]
    [1/2*(2*a*conj(a)+2*y*conj(y)+2*b*conj(b)+2*x*conj(x)-2*(a^2*conj(a)^2-2*a*conj(a)*y*conj(y)+
    2*a*conj(a)*b*conj(b)+2*a*conj(a)*x*conj(x)+y^2*conj(y)^2+2*b*conj(b)*y*conj(y)+
    2*y*conj(y)*x*conj(x)+b^2*conj(b)^2-2*b*conj(b)*x*conj(x)+x^2*conj(x)^2+4*a*conj(x)*y*
    conj(b)+4*b*conj(y)*x*conj(a))^(1/2))^(1/2)]
R15 =
```

[1/2*a+1/2*y+1/2*(a^2-2*a*y+y^2+4*b*x)^(1/2),0]
[0, 1/2*a+1/2*y-1/2*(a^2-2*a*y+y^2+4*b*x)^(1/2)]

6.3.10 代数方程求解

[函数命令]

solve

[调用格式]

f = solve(eq,var)

f 为结果表达式，eq 为方程式 eq=0 左侧的符号表达式，置于单引号内。var 为未知变量（若采用系统默认未知数 x 则没有 var 一项），因此，A1=solve('a*x^2+b*x+c')等价于 A1=solve('a*x^2+b*x+c','x')。

[其他格式]

f = solve(eq1, eq2,…,eqn)

或

f = solve(eq1, eq2,…, eqn,var1,var2,…,varn)

这里，eq1=0，eq2=0，…eq*n*=0 组成了方程组；var1，…，var*n* 为未知变量。

[例 6-16] 求一元二次方程 $ax^2+bx+c=0$ 的根，这里 *a*，*b*，*c* 为系数。

[算例代码]

```
%例 6-16
clear;
syms x a b c;
A1=solve('a*x^2+b*x+c','x')                    %一元二次方程的根，x 未知数
```

[运行结果]

A1 =
[1/2/a*(-b+(b^2-4*a*c)^(1/2))]
[1/2/a*(-b-(b^2-4*a*c)^(1/2))]

[例 6-17] 求方程组 $x^2-ay=0$，$bx=2$ 的解，这里 *a*，*b* 为系数。

```
%例 6-17
clear;
syms x y a b;
[x,y]=solve('x^2-a*y', 'b*x-2','x','y')                %求方程组的解，x，y 为未知数
```

[运行结果]

x =2/b
y =4/a/b^2

重要提示：A1=solve('a*x^2+b*x+c')不能写为 A1=solve('a*x^2+b*x+c=0')；某些方程（尤其是方程中含有周期函数时）使用 solve 命令并不能给出一个区间的全部解，这时可以先画出图形、再利用函数的周期性等特征求出全部解。

6.3.11 微分方程求解

[函数命令]

dsolve

[调用格式]

f = dsolve(eq,cond,v)

f 为结果表达式，表达式中 *C*1、*C*2、*C*3…为积分常数。

eq 为微分方程的符号表达式，置于单引号内，在表达式中使用 Dy 代表 $\mathrm{d}y/\mathrm{d}t$，D2y 代表 $\mathrm{d}^2y/\mathrm{d}t^2$，依此类推，同时不能把 D 作为因变量。

cond 为初始条件，也置于单引号内，如没有给出初始条件，则求得微分方程的通解，此时没有 cond 一项。

v 为自变量，如自变量不确定、则取系统默认变量 *t*，此时没有 *v* 一项。

[其他格式]

f =dsolve(eq1,eq2,…,eqn,cond1,cond2,…,condn)

f =dsolve(eq1,eq2,…,eqn)

这里，eq1，eq2，…，eqn 组成微分方程组，cond1，cond2，…，condn 为初始条件。

[例 6-18] 求微分方程 $\frac{\mathrm{d}y}{\mathrm{d}t}=ay$ 的通解，微分方程 $\frac{\mathrm{d}y}{\mathrm{d}t}=ay$，$y|_{t=0}=1$ 的特解，这里 a 为系数。

[算例代码]

```
%例 6-18
clear;
syms x a b c;
A1=dsolve('Dy=a*y')                                %求微分方程的通解
A2=dsolve('Dy=a*y','y(0)=1')                       %求微分方程的特解
```

[运行结果]

A1 =C1*exp(a*t)

A2 =exp(a*t)

[例 6-19] 求方程组 $\frac{\mathrm{d}x}{\mathrm{d}t}=y$，$\frac{\mathrm{d}y}{\mathrm{d}t}=-x$ 的通解、求解 $\frac{\mathrm{d}^2y}{\mathrm{d}t^2}=a^2y$，$y|_{t=0}=1$，$\left.\frac{\mathrm{d}y}{\mathrm{d}t}\right|_{t=\pi/2}=0$。

[算例代码]

```
%例 6-19
clear;
syms x a b c;
[x,y]=dsolve('Dx=y','Dy=-x')                       %求微分方程组的通解
A2=dsolve('D2y=-a^2*y','y(0)=1','Dy(pi/2)=0')      %求解二阶微分方程
```

[运行结果]

x = C1*sin(t)+C2*cos(t)

y = C1*cos(t)-C2*sin(t)

A2 = sin(1/2*pi*a)/cos(1/2*pi*a)*sin(t*a)+cos(t*a)

6.3.12 数学变换

[函数命令] 表 6-3 为常用的一些数学变换，下面举例说明其用法。

表 6-3 常用数学变换函数

函 数 名	数 学 意 义	调 用 格 式
sinint	正弦积分变换, $Ci(x)=\gamma+In(x)+\int_0^x(\cos t-1)\mathrm{d}t/t$	sinint(X)
cosint	余弦积分变换， $Si(x)=\int_0^x \sin t\mathrm{d}t/t$	cosint(X)
lambertw	拉氏变换, $w(x)\mathrm{e}^{w(x)}=x$	lambertw(X)
hypergeom	高斯变换（超几何变换）, $F(n,d,z)=\sum_{k=0}^{\infty}\frac{C_{n,k}}{C_{d,k}}\frac{z_k}{k!}, C_{v,k}=\prod_{j=1}^{\lvert v\rvert}\frac{\Gamma(v_j+k)}{\Gamma(v_j)}$	hypergeom(n,d,z)
zeta	Z 变换, $\zeta(w)=\sum_{k=1}^{\infty}1/k^w$	zeta(X)

[例 6-20] 计算 1.1 处的正弦积分变换、余弦积分变换与 Z 变换、 $X=\begin{vmatrix}0 & -\mathrm{e}^{-1}\\ \pi & 1\end{vmatrix}$ 时的拉氏变换，以及 n=1，d=2 时的高斯变换（n，d 为变换常数）。

[算例代码]

```
%例 6-20
w1=sinint(1.1)                              %正弦积分变换
w2=cosint(1.1)                              %余弦积分变换
w3=zeta(1.1)                                %Z 变换
w4=lambertw([0 -exp(-1);pi 1])              %拉氏变换
w5=hypergeom(1,2,'z')                       %高斯变换
```

[运行结果]

```
w1 =    1.0287
w2 =    0.3849
w3 =   10.5844
w4 =         0          -1.0000 + 0.0000i
        1.0737            0.5671
w5 =(exp(z)-1)/z
```

习题 6

1．试用 MATLAB 计算 $\sum_{k=0}^{k=\infty}\frac{x^{2k}}{k!}$。

2．试用 MATLAB 分别计算函数或者矩阵函数的极限（a 为常数）。

（1） $X = ax/\sin(x)$，$x \to 0$　　（2） $Y = \begin{vmatrix} 2\sin(x)/x & x/\mathrm{tg}(x) \\ x & x^2 \end{vmatrix}$，$x \to 0$

（3） $f(x) = \left(2 + \dfrac{2}{x}\right)^x$，$x \to +\infty$　　（4） $f(x) = a\left(1 + \dfrac{1}{x}\right)^x$，$x \to +\infty$

3．试用 MATLAB 对函数进行 Taylor 展开。

（1） $f(x) = \cos x$，$x = \pi/2$，到第 5 项　（2）$f(x)=\sin(x)$，$x = \pi/2$，到第 3 项

（3）$y=2/(x-2)$，$x=1.5$，到第 3 项　　（4） $f(x) = \dfrac{x}{\sin x}$，$x = \pi/2$，到第 3 项

4．试用 MATLAB 分别求下列函数或矩阵函数的导数。

（1） $X = ax/\sin(x)$，对 x，三阶导数

（2） $f(x) = \dfrac{1}{a}\left(1 + \dfrac{1}{x}\right)^x$ 对 a，三阶导数

（3） $Y = \begin{vmatrix} 2\sin(x)/x & x/\mathrm{tg}(x) \\ x & x^2 \end{vmatrix}$，对 x，二阶导数

（4） $f(x) = \dfrac{1}{a}\left(1 + \dfrac{1}{x}\right)^x$，对 a，三阶导数

（5） $f(x) = a\dfrac{x}{\sin x}$，对 x，二阶导数

5．试用 MATLAB 计算下列不定积分：

（1） $\displaystyle\int \frac{a}{1+z^2}\mathrm{d}z$，$a$ 为常数　　（2） $\displaystyle\int \frac{xz^2}{1+z^2}\mathrm{d}z$，$x$ 为常数

（3） $\displaystyle\int \frac{x}{1+ax^2}\mathrm{d}a$，$x$ 为常数

6．试用 MATLAB 计算下列定积分。

（1） $\displaystyle\int_0^{3\pi} \sqrt{4\cos^2 t + \sin^2 t}\,\mathrm{d}t$　　（2） $\displaystyle\int_0^1 x\sin(x)\mathrm{d}z$

（3） $\displaystyle\int_{\pi}^{2\pi}\int_{3\pi}^{4\pi} (y\sin x + x\sin y)\,\mathrm{d}x\mathrm{d}y$　　（4） $\displaystyle\int_x^{x+1}\int_0^1 (x^2 + 2y^2 + 1)\,\mathrm{d}x\mathrm{d}y$

7．试用 MATLAB 求下列微分方程的通解（a 为常数）。

（1） $\dfrac{\mathrm{d}y}{\mathrm{d}t} = at$　　（2） $\dfrac{\mathrm{d}y}{\mathrm{d}t} = ay$

（3） $\dfrac{\mathrm{d}y}{\mathrm{d}t} = y + 2$　　（4） $\dfrac{\mathrm{d}x}{\mathrm{d}t} = 2y$，$\dfrac{\mathrm{d}y}{\mathrm{d}t} = -x + 1$

8．试用 MATLAB 求下列微分方程的特解（a 为常数）。

（1） $\dfrac{\mathrm{d}y}{\mathrm{d}t} = at + 1$，$y|_{t=0} = 1$　　（2） $\dfrac{\mathrm{d}^2 y}{\mathrm{d}t^2} = a^2 t$，$y|_{t=0} = 0$，$\left.\dfrac{\mathrm{d}y}{\mathrm{d}t}\right|_{t=\pi/2} = 1$

（3） $\dfrac{\mathrm{d}y}{\mathrm{d}t} = ay$，$y(0) = 1$

9．试用 MATLAB 合并 $(x+y)(\sin(x)+y)$ 中的同类项、化简 $\cos^2 x+\sin^2 x+2$、展开多项式 $\sin(x+y)$、对 x^3-y^3 进行因式分解。

10．试用 MATLAB 求方程组 $x^2-ay=1$，$bx+y=2$ 的解，这里 a，b 为系数。

11．计算矩阵 $\boldsymbol{A}=\begin{bmatrix} a & b \\ c & d \end{bmatrix}$ 的逆。

12．试用 MATLAB 计算 1.3 处的正弦积分变换、余弦积分变换与 Z 变换。

13．一段用 MATLAB 编制的 m 文件，运行时显示该程序出现如下一些问题，请分析说明错误，并写出改正后的完整代码。

[m 文件]

```
clear;
syms x a b;
A1=solve('a*x^2+b*x+c','x');
```

[错误信息]

```
>> run 'F:\MATLABT\ks1.m'
??? Error using ==> run
Error: File: f:\matlabt\ks1.m Line: 3 Column: 10
Missing variable or function
```

第 7 章　图 形 处 理

本章要点：

☑ MATLAB 中图形制作的一般过程；
☑ MATLAB 中的基本作图命令；
☑ MATLAB 中图形格式的设置方法；
☑ MATLAB 中利用图形窗口编辑图形的方法；
☑ MATLAB 中声音与动画处理方法。

7.1　图形制作概述

MATLAB 中进行图形制作，通常采用下面的步骤。

1．准备作图数据

在 MATLAB 中，可以通过下述方法获得作图数据：

（1）编制后缀为.txt 的文本文件，使用 load 命令调入数据；
（2）通过用户自编函数得到作图数据；
（3）通过 MATLAB 系统函数得到作图数据；
（4）根据鼠标在屏幕上的位置，使用命令 ginput 获得相应数据；
（5）在命令窗口或自编函数相应位置，直接输入或使用适当命令输入。

例如，要在[0,2]区间内获得正弦函数中自变量 x 和因变量 y 的数据，可以使用下面的语句：

```
x=0:0.01:2;y=sin(x)
```

这里 0.01 为步长。

2．选定作图窗口与作图区域

在 MATLAB 中要获得作图窗口，可以使用命令 figure，也可以使用 plot 等作图命令。如果使用命令 figure，则创建新的作图窗口；如果没有使用命令 figure，而使用 plot 等作图命令时，MATLAB 依次创建标题为 figure No1、figure No2…的图形窗口。

作图区域如果位于当前作图窗口，可以省略这一步（MATLAB 中的默认设置）；作图区域如果位于当前作图窗口的一个子区域，可以使用命令 subplot。

3．调入作图函数命令

MATLAB 中函数作图命令有很多，使用最多的是 plot 命令。可以在作图命令中直接定制图中线形的格式，也可以单独使用另外命令进行定制。

4．设置图形格式

图形格式的设置，包括如下三个方面：

（1）线形及其标记的设置；

（2）坐标轴范围、坐标轴标识、网格线的设置。

（3）坐标轴标签、图例、文本等方面的设置。

5．输出所制作的图形

上述各步骤中，第 1 和第 3 步必不可少，而其他步骤系统通常都有相应的默认设置。例如，要在[0,2]内作正弦函数的图形可以使用下面的语句：

```
x=0:0.01:2;y=sin(x);plot(x,y)
```

7.2 基本作图命令

7.2.1 图形窗口的创建与控制

1．单个图形窗口的创建与控制

[函数命令]

figure

[调用格式]

figure,figure(n)

该命令创建单个图形窗口，若没有打开图形时执行绘图命令，将创建一个图形窗口；若执行命令前已经打开几个图形窗口，则绘图命令将把图形输出到当前窗口中、并把这个窗口中原来的图形覆盖；*n* 为图形窗口编号。

[相关命令]

可以通过命令 get(n)和 set(n)分别获得和设置第 *n* 个图形窗口的有关属性。

[例 7-1] 作出函数 y=sin(x)在区间[0,10]上的图形。

[算例代码]

```
%例 7-1
clear
x=0:0.01:10;                    %输入自变量 x 的区间范围(自变量数值)
y=sin(x);                       %输入因变量 y 的数值
h=figure(1);                    %创建图形窗口
plot(x,y);                      %绘制由 x，y 构成的图形
get(h);                         %获得图形属性
set(h,'Visible','on');          %设置图形 h 为可见
hold on;                        %保持当前绘图窗口
hold off;                       %释放当前绘图窗口
```

[运行结果] 运行结果如图 7-1 所示，同时在命令窗口显示图形有关属性。

2．多重子图窗口的创建

[函数命令]

subplot

[调用格式]

subplot(m,n,p),subplot(mnp),subplot('position',[l b w h])

subplot 命令将图形窗口分割为多个子图窗口，若执行命令前已经存在某一子图，则该命

令将新图形输出到相应子图，并把原来子图覆盖；*m*，*n* 为子图窗口大小，*m* 为子图行数，*n* 为子图列数；*p* 为子图窗口序号；'position'为 subplot 的位置属性；*l*、*b*、*w*、*h* 分别为子图左下角、底部、宽度、高度位置属性值。

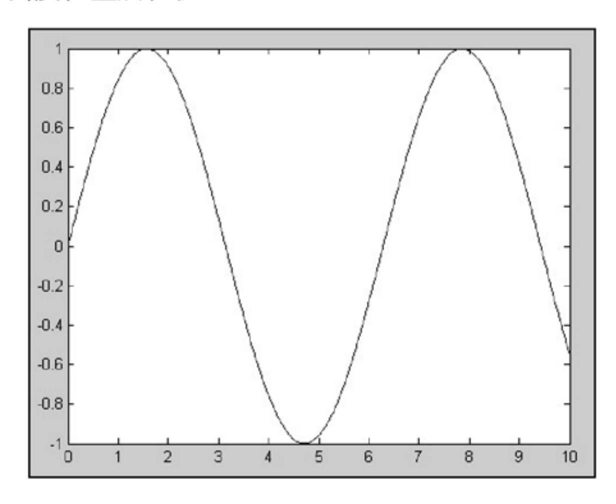


图 7-1　MATLAB 中单个图形窗口的创建

[例 7-2]　试在同一图形窗口、不同坐标系中分别作出 $y=\sin(x)$、$y=\sin(2x)$、$y=\sin(3x)$、$y=\sin(4x)$在 $x\in[0,2\pi]$的图形。

[算例代码]

```
%例 7-2
clear                                              % 清除当前工作区间变量
x=(0:0.01:2)*pi;                                   % 确定自变量值 x
y1=sin(x);y2=sin(2*x);y3=sin(3*x);y4=sin(4*x);     % 确定因变量值 y1-y4
a=subplot(2,2,1);plot(x,y1)                        % 第 1 个子图中作 x-y1 的图形
a=subplot(2,2,2);plot(x,y2)                        % 第 2 个子图中作 x-y2 的图形
a=subplot(2,2,3);plot(x,y3)                        % 第 3 个子图中作 x-y3 的图形
a=subplot(2,2,4);plot(x,y4)                        % 第 4 个子图中作 x-y4 的图形
```

[运行结果]　运行结果如图 7-2 所示。

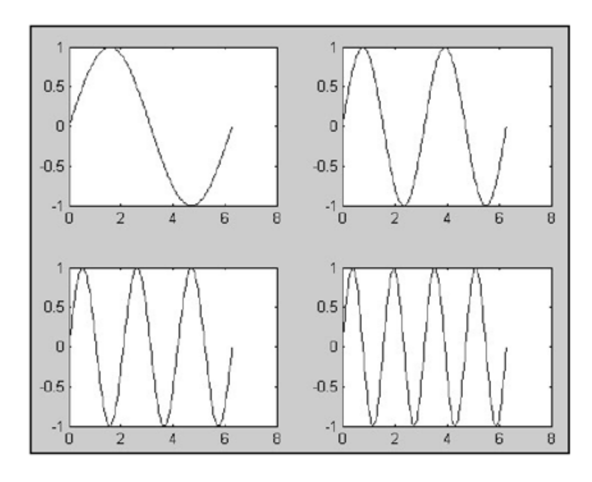


图 7-2　MATLAB 中多个子图窗口的创建

7.2.2 获取图形数据

1．读取图形窗口数据

[函数命令]

ginput

[调用格式]

[x,y]=ginput

ginput 为从鼠标所在位置获得数据；[x,y]为鼠标所在位置坐标。

2．从外部读取图像数据

[函数命令]

imread

[调用格式]

x=imread(filename,fmt),[x,map]=imread(filename,fmt)

imread 为从外部文件 filename 读取格式 fmt 的图像数据，若 F:\MATLABT\lx 为 jpeg 格式的文件，则 filename='F:\MATLABT\ch5\lx',fmt='jpeg'，fmt 可为'jpeg'、'tiff'、'gif'、'bmp'等之一；*x* 为所获取的图像，对于灰度图像和彩色图像，*x* 分别为二维和三维数组；map 为图像信息。

[相关命令]

命令 colormap(map)可以获得图形的颜色信息，而 image(x)则重新绘制图形。

7.2.3 根据数据点作图

1．线形图

（1）直角坐标系中绘图

[函数命令]

plot, plot3

[调用格式]

plot(x),plot(x,y),plot(x,y,s), plot(x1,y1,s1,x2,y2,s2,…)

plot3(x,y,z),plot3(x,y,z,s), plot(x1,y1,z1,s1,x2,y2,z2,s2,…)

plot(x)为绘制向量 *x* 对应的图形，纵坐标为 *x* 的大小，横坐标为数据序号，如果 *x* 为复数，则纵坐标为 *x* 虚部、横坐标为 *x* 实部。

plot(x,y)绘制数据 *x–y* 对应的图形。

plot(x,y,s)以格式 *s* 绘制数据 *x–y* 对应的图形，这是 MATLAB 中绘制图形的最基本命令。

plot(x1,y1,s1,x2,y2,s2,…)为以格式 *s*1，*s*2…绘制数据 *x*1–*y*1，*x*2–*y*2…对应的图形。

s，*s*1，*s*2…为颜色、线型、点型的格式。

plot3 是与 plot 对应的三维曲线作图命令，使用方法与 plot 一致，只是数据点对（*x*,*y*）改为数据点对（*x*,*y*,*z*），可用于二元函数的作图。

[例 7-3] 已知 *x*=[1.2 7.0 3.6 5.0 8.0]，*y*=[4.1 5.2 6.3 9.0 15.0]，z=[11.1 15.2 16.3 19.0 25.0]，试绘制 *x–y* 对应的图形与 *x–y–z* 对应的图形。

[算例代码]

```
%例 7-3
clear                                    % 清除当前工作区间变量
x=[1.2 7.0 3.6 5.0 8.0];                 % 确定变量值 x
y=[4.1 5.2 6.3 9.0 15.0];                % 确定变量值 y
z=[11.1 15.2 16.3 19.0 25.0];            % 确定变量值 z
a=subplot(1,2,1);plot(x,y)               % 第 1 个子图以默认格式作 x-y 的图形
a=subplot(1,2,2);plot3(x,y,z)            % 第 2 个子图以默认格式作 x-y-z 的图形
```

[运行结果] 见图 7-3。

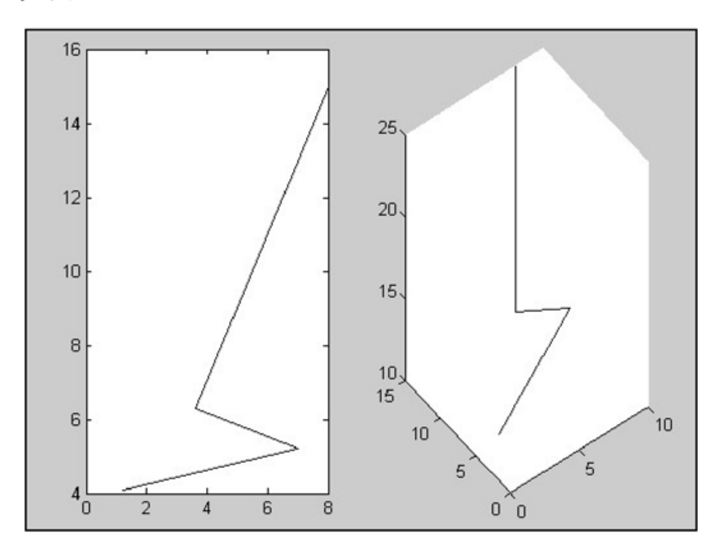


图 7-3　MATLAB 中根据数据点绘制线形图

（2）对数坐标系中绘图

[函数命令]

loglog,semilogx,semilogy

[调用格式]

loglog(x,y), loglog(x,y,s), loglog(x1,y1,s1,x2,y2,s2,…),

semilogx(x,y), semilogx(x,y), semilogx(x1,y1,s1,x2,y2,s2,…),

semilogy(x,y), semilogy(x,y), semilogy(x1,y1,s1,x2,y2,s2,…)

loglog(x,y)为绘制数据 *x–y* 对应的图形，图中纵横坐标均取对数。

loglog(x,y,s)为以格式 *s* 绘制数据 *x–y* 对应的图形，纵横坐标均取对数。

loglog(x1,y1,s1,x2,y2,s2,…)以格式 *s*1，*s*2…绘制数据 *x*1–*y*1，*x*2–*y*2…对应的图形，纵横坐标均取对数。

semilogx(…)，semilogy(…)与 loglog 命令的使用方法基本一致，只是分别取横坐标为对数、纵坐标为对数。

s，*s*1，*s*2…为颜色、线型、点型的格式。

要注意的是，log10(0)数学上没有意义，但 MATLAB 并不给出错误信息。

[例 7-4]　已知 *x*=[1.2 7.0 3.6 5.0 8.0]，*y*=[4.1 5.2 6.3 9.0 15.0]，*z*=[11.1 15.2 16.3 19.0 25.0]，试分别在双对数坐标系、半对数坐标系（分别取纵横轴为对数）绘制 *x–y* 对应的图形。

[算例代码]

```
%例 7-4
clf                                   %清除当前图形
x=[1.2 7.0 3.6 5.0 8.0];              % 确定变量值 x
y=[4.1 5.2 6.3 9.0 15.0];             % 确定变量值 y
a=subplot(1,3,1);loglog(x,y)          %子图 1:双对数坐标系作 x-y 曲线
a=subplot(1,3,2);semilogx(x,y)        %子图 2:半对数坐标系(x 取对数)作 x-y 曲线
a=subplot(1,3,3);semilogy(x,y)        %子图 3:半对数坐标系(y 取对数)作 x-y 曲线
```

[运行结果]

运行结果如图 7-4 所示。

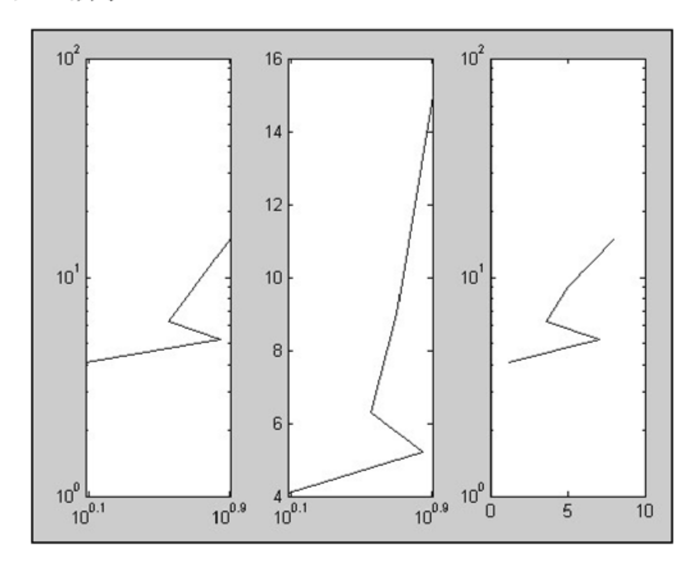


图 7-4 MATLAB 中根据数据点在对数坐标系中绘图

（3）极坐标系中绘图

[函数命令]

polar

[调用格式]

polar(theta,r), polar(theta,r,s), polar(theta1,r1,s1,theta2,r2,s2,…)

polar(theta,r)为极坐标系中绘制数据 theta-r 对应的图形；polar(theta,r,s)为以格式 s 在极坐标系中绘制数据 theta-r 对应的图形；polar(theta1,r1,s1,theta2,r2,s2,…)以格式 s1,s2,…在极坐标系中绘制数据 theta1-r1,theta2-r2,…对应的图形；theta，r 分别为极径和角度（弧度）；*s*，s1，s2…为颜色、线型、点型的格式。

重要提示：可使用命令[x,y]=pol2cart(theta,r)将极坐标系的数据点对(theta,r)转化为直角坐标系的数据点对[x,y]，命令 plot(x,y)和命令 polar(theta,r)的效果相同。

[例 7-5] 分别在极坐标系和直角坐标系中作出三叶玫瑰线 $r = a\cos 3\theta$ 的图形，这里 a=2。

[算例代码]

```
%例 7-5
clf                                   %清除当前图形
```

```
a=2;                                  %参数a取2
theta=(0:0.01:4)*pi;                  %计算角度
r=a*cos(3*theta);                     %计算极径
subplot(1,2,1);polar(theta,r);        %在极坐标系中绘制三叶玫瑰线
[x,y]=pol2cart(theta,r);              %将极坐标数据转化为直角坐标系数据
subplot(1,2,2);plot(x,y);             %在直角坐标系中绘制三叶玫瑰线
axis equal;                           %坐标轴纵横比例尺相同
```

[运行结果] 运行结果如图 7-5 所示。

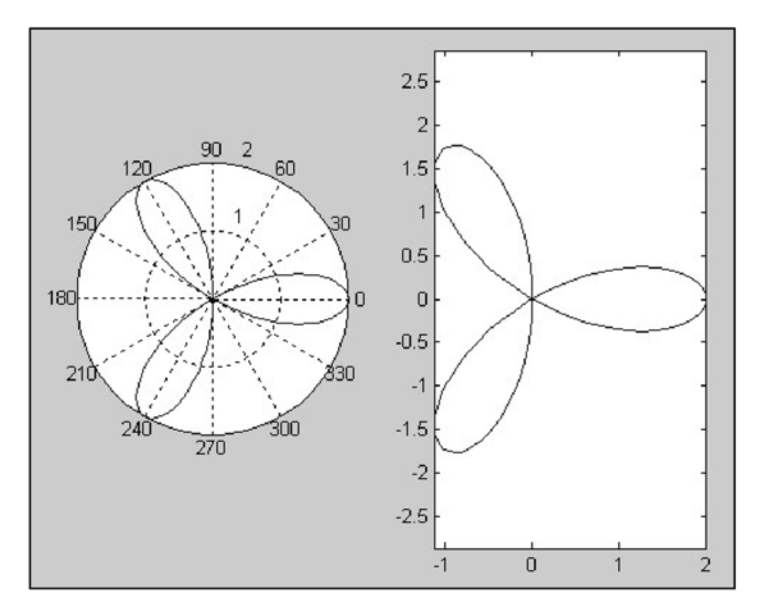


图 7-5 MATLAB 中根据数据点在极坐标系中绘图

（4）双轴图

[函数命令]

plotyy

[调用格式]

[haxes,hline1,hline2]=plotyy(x1,y1,x2,y2,m1,m2)

plotyy(x1,y1,x2,y2,m1,m2)为分别以作图命令 *m1* 与 *m2* 绘制数据 *x1*−*y1* 与数据 *x2*−*y2* 对应的图形；haxes,hline1,hline2 分别为坐标轴对象，线形图 1 对象，线形图 2 对象。

[例 7-6] 分别作出 $z_1 = Ae^{-at}$ 与 $z_2 = \sin(bt)$ 的双轴图，这里 $t \in [0,900]$，A=1000，a=b=0.005。

[算例代码]

```
%例 7-6
clf                                          %清除当前图形
t = 0:900;A = 1000;a = 0.005;b = 0.005;      %确定参数 t，A，a，b
z1 = A*exp(-a*t);                            %确定参数 z1
z2 = sin(b*t);                               %确定参数 z2
%以半对数和直角坐标系绘制 t-z1 和 t-z2 曲线，并获得有关句柄对象
[haxes,hline1,hline2] = plotyy(t,z1,t,z2,'semilogy','plot');
```

[运行结果] 运行结果如图 7-6 所示。

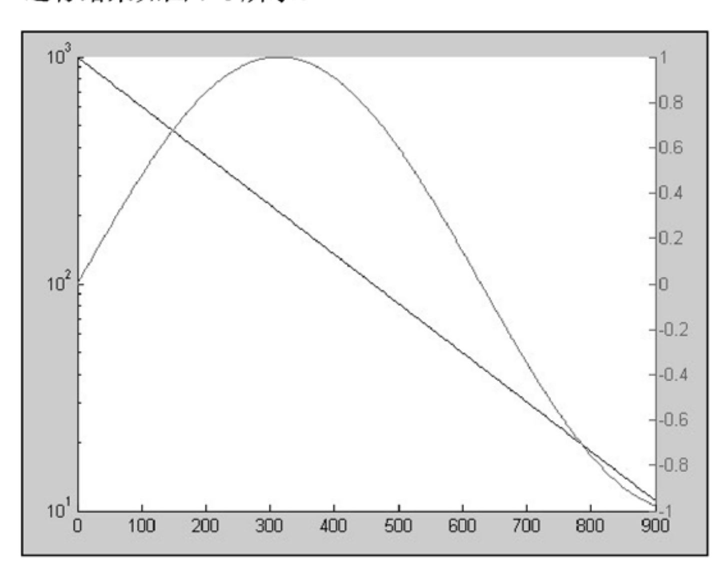


图 7-6 MATLAB 中根据数据点绘制双轴图

2．其他图形

[函数命令]

bar,bar3,pie,pie3,stem,stem3,quiver,feather,compass,rose,

contourf,area,hist,gplot,stairs,errorbar,comet

[调用格式]

bar(x),bar(y,x),bar3(y),bar3(y,x),pie(x),pie3(x),stem(x),stem3(x),

quiver(x,y,u,v),quiver(u,v),feather(u,v),compass(u,v),rose(theta),

contour(Z),contourf(Z),contour(Z,n),contourf(Z,n),area(x,y),

hist(y),gplot(A,C),stairs(x),errorbar(x,y,l),comet(x,y)

bar(x)为绘制矩阵 ***y***的条形图（直方图）。

bar(y,x)为绘制矩阵 ***y***的条形图（直方图），*x*为单向递增的向量。

bar3(x)为绘制矩阵 ***y***的三维条形图（直方图）。

bar3(y,x)为绘制矩阵 ***y***的三维条形图（直方图），*x*为单向递增的向量。

pie(x)为绘制向量 *x*的饼图。

pie3(x)为绘制向量 *x*的三维饼图。

stem(x)为绘制向量 *x*的针状图。

stem3(x)为绘制向量 *x*的三维针状图。

quiver(x,y,u,v)为绘制向量场图，点（*x,y*）处的箭头由分量（*u,v*）定义，四个矩阵 ***x***，***y***，***u***，***v***的大小必须相等。

quiver(x,y,u,v,s)为绘制向量场图，点（*x,y*）处的箭头由分量（*u,v*）定义，四个矩阵 ***x***，***y***，***u***，***v***的大小必须相等，线形由 s 定义。

feather(u,v)为绘制向量沿横轴等距离分布的羽状图，箭头大小由（*u,v*）定义。

compasss(u,v)为绘制箭头发自圆心的罗盘图，箭头大小由（*u,v*）定义。

rose(theta)为绘制玫瑰花图，theta 为每个花瓣的角度（以弧度表示），而花瓣的长度为 theta

落在该范围内的角度。

contour(Z,n)为绘制矩阵 ***Z*** 的等值线图，*n* 为等值水平数，*n* 为空（即 contour(Z)）则采用系统默认值。

contourf(Z,n)为绘制矩阵 ***Z*** 的等值线图，用法与 contour 类似，但等值区域被填充。

area(x,y)为绘制数据（*x*,*y*）的面积图，用法与 plot 类似，但数值 *y* 区域被填充。

hist(y)为绘制关于数据 *y* 的柱状图。

gplot(A,C)为绘制数据（A,C）的拓扑图，***A*** 为邻接矩阵，***C*** 为 ***A*** 的相配矩阵。

stairs(x)为绘制向量 *x* 的阶梯图。

errorbar(x,y,l)为绘制向量 *y* 对向量 *x* 的误差条形图，长度为 l 的误差条对称地分布在 *y* 的上下方。

comet(x,y)为绘制向量 *y* 对向量 *x* 的彗星图。

MATLAB 给出这些图形的很多格式，上述只是最常用的一些形式，读者可以通过命令窗口的帮助命令（help）或 MATLAB 自带的自述文件学习自身专业领域所用的图形格式及其相应设置，下面给出常用形式的一些例子。

[例 7-7]　绘制 ***x***=[1 2 3 4; 2 3 4 5;3 4 5 6]，***y***=[1 4 7]的条形图。

[算例代码]

```
%例 7-7
clf                              %清除当前图形
x=[1 2 3 4; 2 3 4 5;3 4 5 6];    %输入数据 x
y=[1 4 7];                       %输入数据 y
subplot(2,2,1),bar(x);           %子图 1：关于 x 的条形图
subplot(2,2,2),bar(y,x);         %子图 2：关于(y,x)的条形图
subplot(2,2,3),bar3(x);          %子图 3：关于 x 的三维条形图
subplot(2,2,4),bar3(y,x);        %子图 4：关于(y,x)的三维条形图
```

[运行结果]

运行结果如图 7-7 所示。

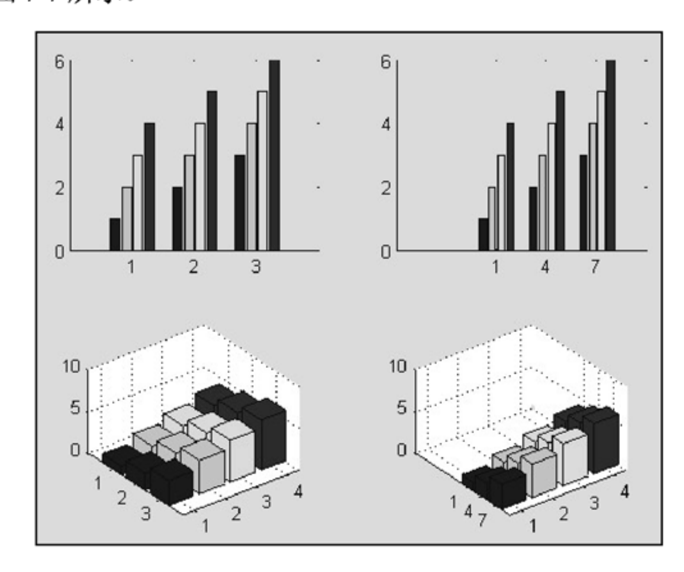


图 7-7　根据数据点绘制条形图

[例 7-8] 绘制 *x*=[1 2 3 4 5 6]的饼图与针状图。

[算例代码]

```
%例 7-8
clf                                  %清除当前图形
x=[1 2 3 4 5 6];                     %输入数据 x
subplot(2,2,1),pie(x);               %子图 1：关于 x 的饼图
subplot(2,2,2),pie3(x);              %子图 2：关于 x 的三维饼图
subplot(2,2,3),stem(x);              %子图 3：关于 x 的针状图
subplot(2,2,4),stem3(x);             %子图 4：关于 x 的三维针状图
```

[运行结果] 执行结果见图 7-8。

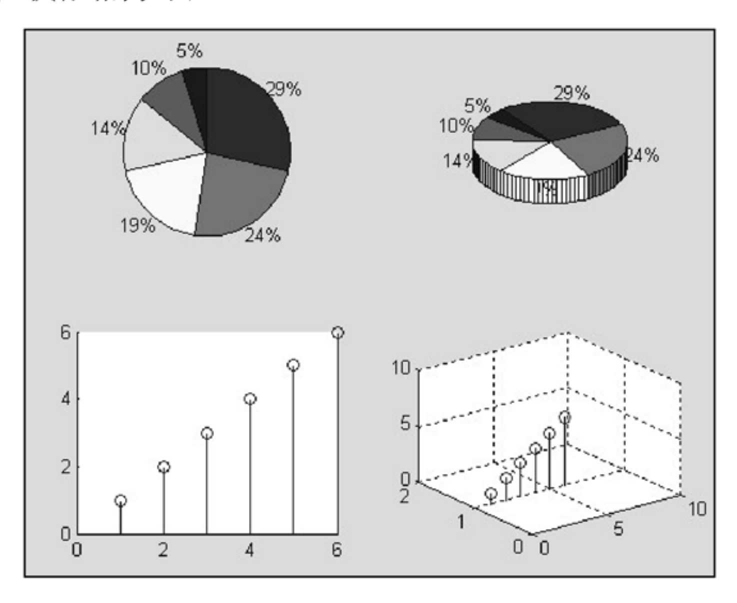


图 7-8 根据数据点绘制饼图与针状图

[例 7-9] 绘制 *x*=[1 2 3 4 5 6]，*y*=[1 2 3 4 5 6]，*u*=[1 2 3 4 5 6]，*v*=[1 2 3 4 5 6]的蓝色和红色向量场图及关于（*u,v*）的羽状图、罗盘图。

[算例代码]

```
%例 7-9
clf                                        %清除当前图形
x=[1 2 3 4 5 6];y=[1 2 3 4 5 6];           %输入数据 x,y
u=[1 2 3 4 5 6];v=[1 2 3 4 5 6];           %输入数据 u,v
subplot(2,2,1),quiver(x,y,u,v);            %子图 1：关于(x,y,u,v)的向量场图
subplot(2,2,2),quiver(x,y,u,v,'r');        %子图 2：(x,y,u,v)的红色向量场图
subplot(2,2,3),feather(u,v);               %子图 3：(u,v)的羽状图
subplot(2,2,4),compass(u,v);               %子图 3：(u,v)的罗盘图
```

[运行结果]

运行结果如图 7-9 所示。

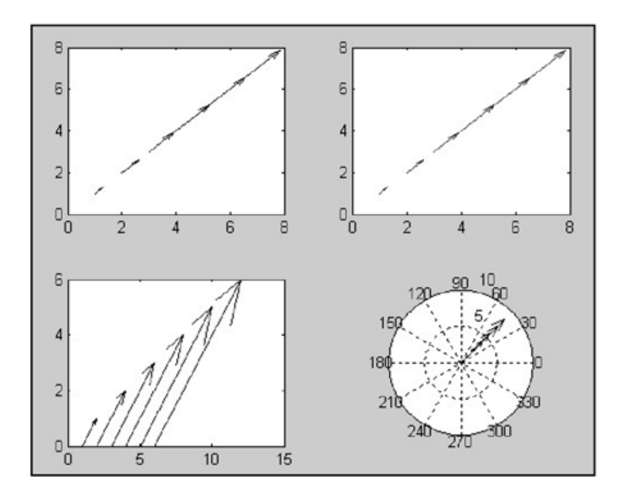


图 7-9　根据数据点绘制向量场图、羽状图与罗盘图

[例 7-10]　绘制 50 个随机数据分布特征的玫瑰花图与函数 y=sin(x)在 $x\in[0,2\pi]$中的面积图，根据峰值函数 peaks 绘制等值线图，并作出 1000 个随机数的柱状图。

[算例代码]

```
%例 7-10
clf                                         %清除当前图形
theta=10*rand(1,50);                        %确定 50 个随机数 theta
Z=peaks;                                    %确定 Z 为峰值函数 peaks
x=0:0.01:2*pi;y=sin(x);                     %确定正弦函数数据点 x,y
t=randn(1000,1);                            %确定 1000 个随机数 t
subplot(2,2,1),rose(theta);                 %子图 1:关于(theta)的玫瑰花图
subplot(2,2,2),area(x,y);                   %子图 2:关于(x,y)的面积图
subplot(2,2,3),contour(Z);                  %子图 3:关于 Z 的等值线图(未填充)
subplot(2,2,4),hist(t);                     %子图 4:关于 t 的柱状图
```

[运行结果]

运行结果如图 7-10 所示。

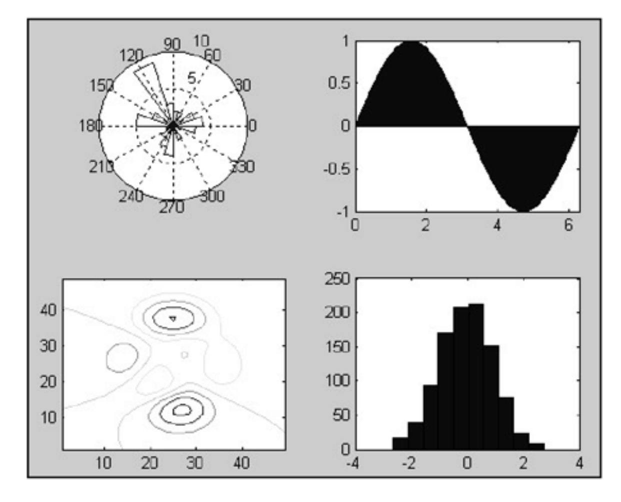


图 7-10　根据数据点绘制玫瑰花图、面积图、等值线图与柱状图

[例 7-11] 绘制由 bucky 函数所产生稀疏邻接矩阵的拓扑图及关于向量 t=[1 2 3 4 5 6]的阶梯图，并在 $x\in[0,4\pi]$上绘制 $y=xe^{\sin x}$的误差条形图与彗星图。

[算例代码]

```
%例 7-11
clf                                          %清除当前图形
[A,C]=bucky;                                 %使用 bucky 函数产生一稀疏邻接矩阵
t=[1 2 3 4 5 6];                             %确定向量 t
x=0:0.7:4*pi;y=x.*exp(sin(x));l=0.1*y;       %确定 x,y 及误差条长度
subplot(2,2,1),gplot(A,C);                   %子图 1：关于[A,C]的拓扑图
subplot(2,2,2),stairs(t);                    %子图 2：关于 t 的阶梯图
subplot(2,2,3),errorbar(x,y,l);              %子图 3：关于(x,y)的误差条形图
subplot(2,2,4),comet(x,y);                   %子图 4：关于(x,y)的彗星图
```

[运行结果] 运行结果如图 7-11 所示。

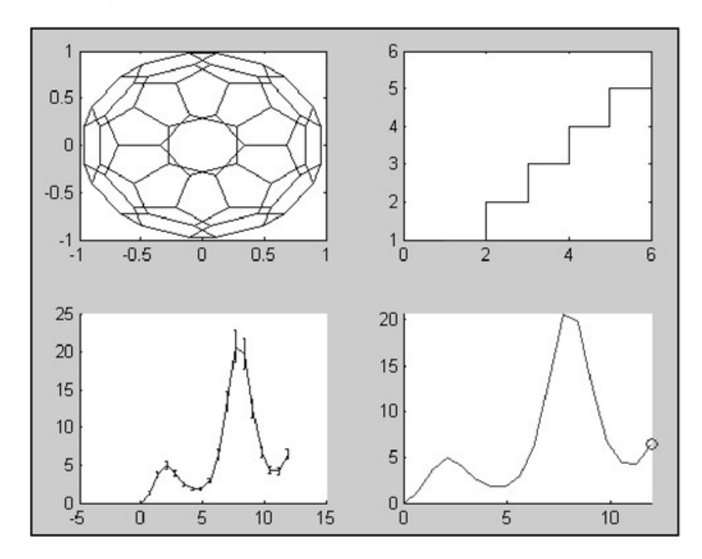


图 7-11 根据数据点绘制拓扑图、阶梯图、误差条形图与彗星图

7.2.4 函数作图

1．平面曲线作图

[函数命令]

fplot,ezplot

[调用格式]

fplot(fun,lims,tol,n,p1,p2,⋯)

ezplot(fun,lims,fig)

fplot(fun,lims,tol,n,p1,p2,⋯)为绘制函数 fun 的图形。

lims 为作图区间，若 lims=[xmin xmax]，则自变量 x 的区间为 $x\in[xmin, xmax]$，若 lims=[xmin xmax ymin ymax]，则 x 与 y 的区间分别为 $x\in[xmin, xmax]$ 与 $y\in[ymin, ymax]$。

tol 为相对误差，若为空，则取默认值 2e–3。

n 为作图点数（即图形点数至少为 n+1 个），若为空，则取默认值 n=1。

$p1,p2\cdots$为函数 fun 的参数，若为空，则没有参数。

ezplot 的用法与 fplot 基本一致，只是 ezplot 可作隐函数与多元函数的图形、默认作图区间为 $x\in[-2\pi,2\pi]$、使用了一些图形标记，此外，ezplot 还可作隐函数形式 $f(u,v)=0$ 的矢量图。

fig 为制图窗口句柄。

[例 7-12] 使用命令 fplot 作出函数 $y=\sin(x)$ 的图形，自变量区间分别为：$x\in[0,2\pi]$；$x\in[\pi,3\pi]$；$x\in[-\pi,\pi]$，$y\in[-\pi,\pi]$；$x\in[0.01,0.1]$（此时要求作图误差小于 0.001）。

[算例代码]

```
%例 7-12
clf                                                  %清除当前图形
subplot(2,2,1),fplot('sin(x)',[0 2*pi]);             %子图 1：[0 2*pi]内作函数图形
subplot(2,2,2),fplot('sin(x)',[pi 3*pi]);            %子图 2：[pi 3*pi]内作函数图形
subplot(2,2,3),fplot('sin(x)',pi*[-1 1 -1 1]);       %子图 3：xy 为一定区间的函数图形
subplot(2,2,4),fplot('sin(x)',[ 0 2*pi],1e-3);       %子图 4：具有样误差控制的函数图形
```

[运行结果]

运行结果如图 7-12 所示。

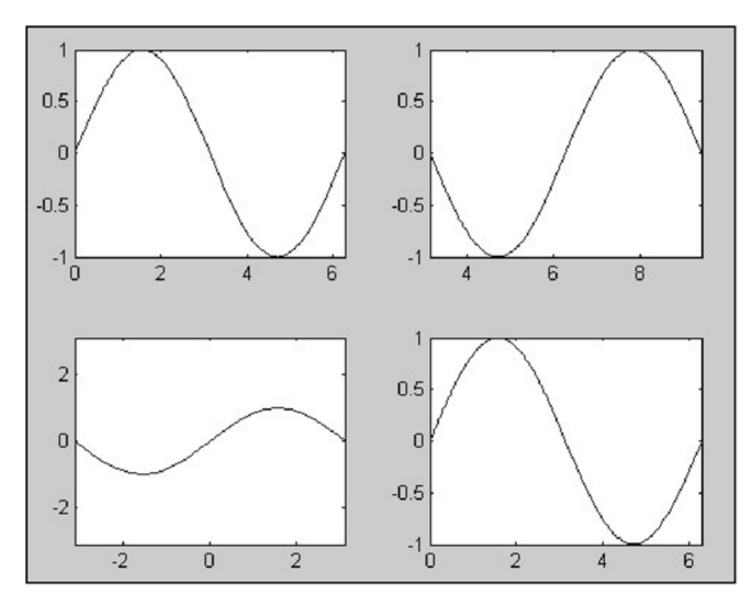


图 7-12　根据数据点使用命令 fplot 绘制平面曲线图

[例 7-13] 绘制函数 $u^2+v^2-1=0$ 在 $u\in[-3,2]$，$v\in[-2,3]$ 上的图形。

[算例代码]

```
%例 7-13
clf                                      %清除当前图形
ezplot('u^2-v^2-1',[-3 2 -2 3]);         % u^2-v^2-1＝0 的矢量作图
```

[运行结果] 运行结果如图 7-13 所示。

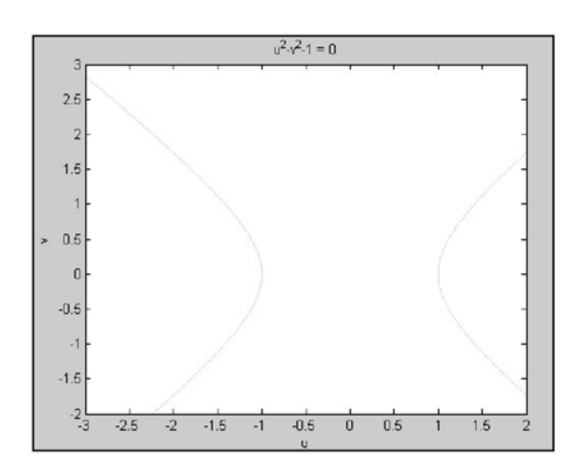


图 7-13　根据数据点使用命令 ezplot 绘制平面曲线图

2．二元函数曲线作图

二元函数曲线作图可以使用命令 plot3，其使用方法前已述及，下面给出一个二元函数曲线作图的例子。

[例 7-14]　试在区间 $x\in[0,2\pi]$, $y\in[0,2\pi]$ 作出 $z=\cos x+\sin y$ 对应的曲线。

[算例代码]

```
%例 7-14
clf                                      %清除当前图形
x=0:0.01:2*pi;y=0:0.01:2*pi;             %确定变量 x,y
z=cos(x)+sin(y);                         %确定变量 z
plot3(x,y,z);                            %三维空间绘制曲线
```

[运行结果]

运行结果如图 7-14 所示。

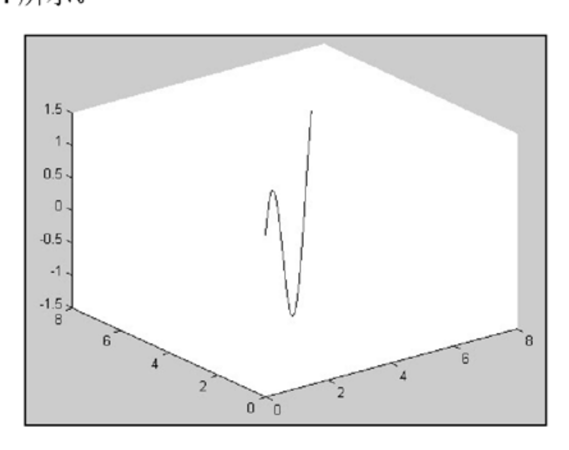


图 7-14　根据数据点使用命令 plot3 绘制二元函数曲线图

7.2.5　三维图形制作

[函数命令]

meshgrid, mesh,surf,meshc,surfc,meshz

[调用格式]

[x y]=meshgrid(x,y),

mesh(x,y,z),surf(x,y,z),meshc(x,y,z),surfc(x,y,z),meshz(x,y,z)

[x y]=meshgrid(x,y)为根据已有 m 个数据的向量 x 和 n 个数据的向量 y 分别生成有 $m \times n$ 个数据的新矩阵 $\boldsymbol{x}$ 和 $\boldsymbol{y}$。

z 为关于 x，y 的函数。

mesh(x,y,z)为绘制数据(x,y,z)的三维网格表面。

surf(x,y,z)为绘制数据(x,y,z)的三维曲面。

meshc(x,y,z)为绘制(x,y,z)的三维网格表面，并在网线图下添加等值线。

surfc(x,y,z)为绘制数据(x,y,z)的三维曲面并在图下添加等值线。

meshz(x,y,z)为绘制(x,y,z)的三维网格表面，并在网线图下添加零平面。

重要提示：三维图形制作通常是在确定向量 x，y 的基础上，使用命令 meshgrid 生成新的矩阵，再输入函数 $z=f(x,y)$，最后使用 mesh 等命令生成三维网格、使用 surf 等命令生成三维曲面。

[例 7-15]　在 $x \in [-5,5]$，$y \in [-4,4]$ 上作出 $z=\frac{1}{2}x^3+y^2$ 对应的三维网格表面和三维曲面。

[算例代码]

```
%例 7-15
clf                                % 清除当前图形
x=-5:0.1:5;y=-4:0.1:4;             % 确定变量 x，y 的范围（向量）
[x,y]=meshgrid(x,y);               % 生成变量 x，y 的网格点（矩阵）
z=0.5*x.^3+y.^2;                   % 生成变量 z
subplot(2,2,1);mesh(x,y,z);        % 子图 1：三维网格表面
subplot(2,2,2);surf(x,y,z);        % 子图 2：三维曲面
subplot(2,2,3);meshc(x,y,z);       % 子图 3：三维网格表面（具有等值线）
subplot(2,2,4);surfc(x,y,z);       % 子图 4：三维曲面（具有等值线）
```

[运行结果]

运行结果如图 7-15 所示。

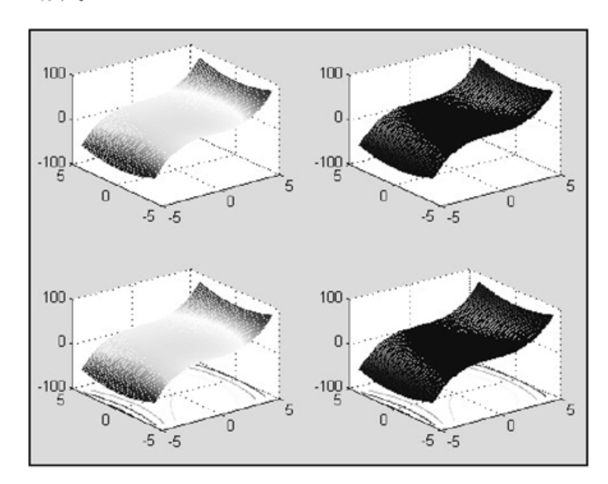


图 7-15　根据数据点使用 mesh 等命令 plot3 绘制三维图形

7.3　图形格式的设置

7.3.1　线性图格式的设置

在类似 plot(x,y,s)形式的作图中，格式 *s* 可以使用系统默认的格式，也可以重新设置，常用的格式 *s* 见表 7-1，其他格式（比如线宽等）可以通过 help plot 学习使用。如以线宽 5、红色、点划线、叉号形式绘制 $x\in[0,2\pi]$ 内的正弦函数 $y=\sin x$ 的图形，可以使用下述命令：

```
x = 0:pi/10:2*pi;y= sin(x);plot(x,y,'r-.x','LineWidth',5);
```

表 7-1　线形图格式的的设置

	字　符	颜　色
颜色的定制	b	蓝色
	c	青色
	g	绿色
	k	黑色
	m	洋红色
	r	红色
	w	白色
	y	黄色
线型的定制	字　符	线　型
	-	实线
	:	点线
	-.	点划线
	--	虚线
点型的定制	字　符	点　型
	.	点号
	o	圆圈号
	x	叉号
	+	十字形号
	*	星号
	s	方块号
	d	钻石形号
	p	五角星号
	h	六角星号
	V	顶点向下的三角形
	<	顶点向左的三角形
	>	顶点向右的三角形
	^	顶点向上的三角形

如果只作数据点，格式 *s* 直接设置为点型即可，如只作出数据点：

```
x=[0 1 2 3];y=[2 3 4 5];plot(x,y,'o');
```

而作成系统默认的实线：

```
x=[0 1 2 3];y=[2 3 4 5];plot(x,y);
```

7.3.2　图形标签、图例和文本的设置

1．图形与坐标轴的删除

[函数命令]

cla,clf

[调用格式]

cla,clf

clf 为清除当前图形；cla 为清除当前坐标轴。

2．坐标轴定义与设置

[函数命令]

axis,clf axis equal

[调用格式]

axis(lims),axis equal

axis 为坐标轴范围；lims=[xmin xmax ymin ymax]为作图时 *x* 的最小值和最大值、y 的最小值和最大值；axis equal 为设置坐标轴纵横比例尺相同。

3．网格线设置

[函数命令]

grid on, grid off

[调用格式]

grid on, grid off

grid on 为增加网格线；grid off 为删除网格线。

4．图例设置

[函数命令]

legend, legend off

[调用格式]

legend(str1,str2,…,k), legend off

legend(str1,str2,…,k)为增加图例。str1,str2…为图例标题，与图形内曲线依次对应。*k* 为图例放置位置的参数：

- *k*=−1，放在图形右侧；
- *k*=0，放在图内“最好”位置（覆盖数据最少）；
- *k*=1（默认值），放在图内右上角；
- *k*=2，放在图内左上角；
- *k*=3，放在图内左下角；
- *k*=4，放在图内右下角。

legend off 为删除图例。

5．标题设置

[函数命令]

title, xlabel, ylabel,zlabel

[调用格式]

title(str), xlabel(str), ylabel(str), zlabel(str)

title 为图形标题；xlabel，ylabel，zlabel 分别 *x*，*y*，*z* 轴的标题（*z* 轴标题只在三维图形中生效)；str 为标题字符串，也可为结构数组（如为结构数组则分行显示数组元素)。

6．文本设置

[函数命令]

text, gtext

[调用格式]

text(x,y,str), gtext(str)

text(x,y,str)为位置(*x*,*y*)处增加文本 str；gtext(str)为鼠标位置处增加文本 str。

7．双轴图两侧标记处理

plotyy 命令绘制的双轴图有两个坐标系，可通过图形对象属性设置对两侧标记进行处理。例如绘制双轴图：

```
x=0:0.01:2*pi;y1=sin(x);y2=cos(x);[AX,H1,H2]=plotyy(x,y1,x,y2,'plot');
```

要将其左、右坐标轴 AX(1)和 AX(2)标题分别设置为“正弦函数”与“余弦函数”，则使用语句：

```
set(get(AX(1),'Ylabel'),'string','正弦函数'); set(get(AX(2),'Ylabel'),'string','余弦函数');
```

若要将两根曲线 H1 和 H2 的颜色分别设置为红色和蓝色，则使用语句：

```
set(H1,'Color','r'); set(H2,'Color','b');
```

7.3.3 增加图形元素

[函数命令]

hold on, hold off

[调用格式]

hold on, hold off

“hold on”为保持已作图形，以便在同一图形窗口内作下一图形；“hold off”为释放已作图形。

[例 7-16] 清除已有图形，在同一坐标系中 $x\in[0,2\pi]$ 绘制分别以红实线和蓝虚线作出 $y_1=\sin x$，$y_2=\cos x$ 对应的曲线，并设置适当的图例与标题。

[算例代码]

```
%例 7-16
clf                                          %清除当前图形
x=(0:0.01:2)*pi;y1=sin(x);y2=cos(x);         %确定数据点 x,y1,y2
plot(x,y1,'r-');                             %以红实线绘制 x-y1
hold on;                                     %保持曲线，以便作下一曲线
plot(x,y2,'b--');                            %以蓝虚线绘制 x-y2
grid on;                                     %增加网格线
legend('正弦曲线','余弦曲线',1);              %设置不同曲线的图例
title({'函数曲线的绘制','格式设置示例'});      %使用结构数组设置图形标题
xlabel('横轴');ylabel('纵轴');                %设置纵横轴标题
```

[运行结果]

运行结果如图 7-16 所示。

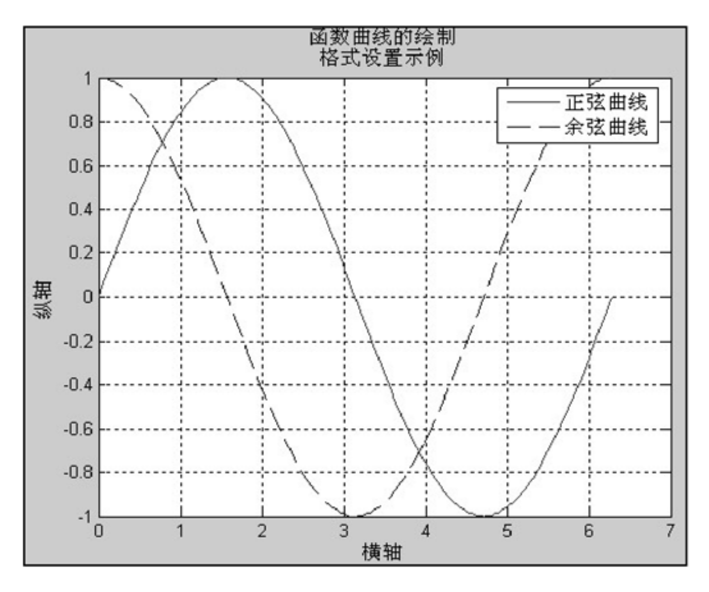


图 7-16　图形格式的设置

7.3.4　色图处理

[函数命令]

colormap,colorbar,brighten

[调用格式]

colormap,colormap(map),brighten(s)

colormap 为当前色图表的矩阵。

colormap(map)为将当前图形色图设置为系统预定义的 map 格式，map 常为下列格式之一：hsv（红色-蓝色-深红色），gray（线性灰色），pink（粉红色），cool（青色-深红色），automun（红色-黄色），spring（红紫色-黄色），winter（蓝色-绿色），summer（绿色-黄色）。

brighten(s)为改变当前图形窗口色图的亮度，若$-1<s<0$ 则加暗当前色图，若 $0<s<1$ 则增亮当前色图。

[例 7-17]　试将 $x\in[-5,5]$， $y\in[-4,4]$ 上三维网格表面 $z=(x^3+y^2)/2$ 的色图设置为红色-蓝色-深红色色图，同时将图形适当增亮。

[算例代码]

```
%例 7-17
clf                                   % 清除当前图形
x=-5:0.1:5;y=-4:0.1:4;                % 确定变量 x，y 的范围（向量）
[x,y]=meshgrid(x,y);                  % 生成变量 x，y 的网格点（矩阵）
z=0.5*x.^3+y.^2;                      % 生成变量 z
mesh(x,y,z);                          % 绘制三维网格曲面
colormap(hsv);                        %设置三维网格曲面色图
brighten(0.60);                       %加亮当前图形
```

[运行结果]

运行结果如图 7-17 所示。

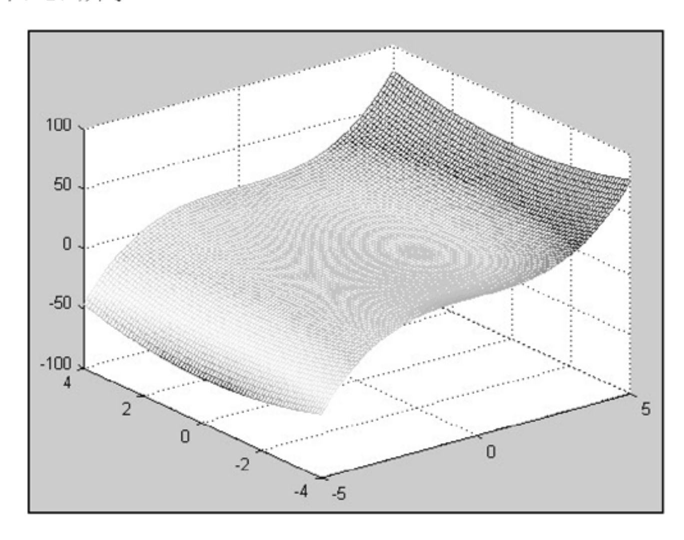


图 7-17 色图格式与颜色刻度的设置

7.4 利用图形窗口编辑图形

图形窗口是指 MATLAB 中使用绘图函数和工具进行图形绘制的一个单独窗口。利用图形窗口编辑图形，实际上是交互绘图，前面介绍的有关图形制作命令，都可以利用 MATLAB 的图形窗口交互完成。

MATLAB 的图形窗口界面如图 7-18 所示，其从上到下分为 4 个部分：图形窗口标题栏、菜单栏、快捷工具栏、图形显示窗口。菜单栏是图形窗口控制图形的主要部分，其组成如图 7-19 所示。下面简述各菜单栏的主要用法。

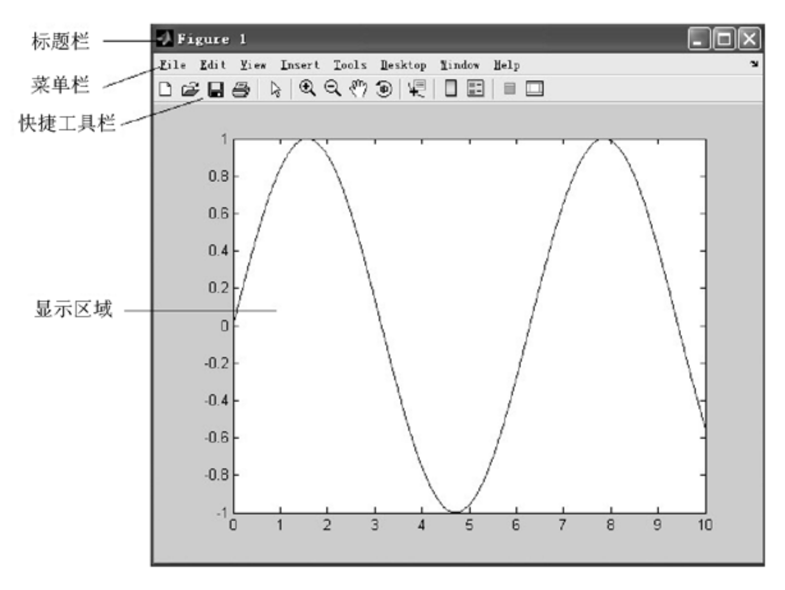


图 7-18 图形窗口

1．File 菜单

File 菜单如图 7-19 所示。

New ▶M-File/Figure/Model/Variable/Gui：创建新的 m 文件、图形、模型、变量、图形用户界面。

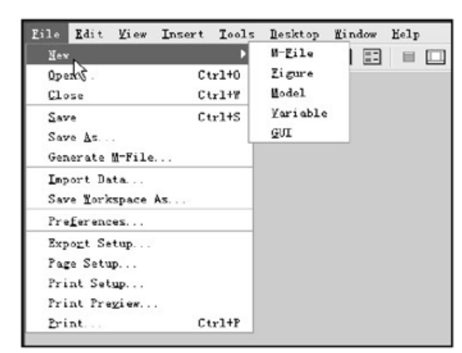


图 7-19　File 菜单

Open：弹出对话框，打开已有图形。

Close：关闭当前图形窗口。

Save：如未命名当前图形则弹出对话框进行保存，否则保存当前图形。

Save As：弹出对话框，将当前图形另存。

Generate M-File：编写 m 文件。

Import Data：载入数据文件。

Save Workspace As：弹出对话框，将当前工作区间变量另存。

Preferences：弹出对话框，对 MATLAB 平台有关内容进行设置。

Export Setup：弹出对话框，将当前图形的另存图形格式进行设置。

Page Setup：弹出对话框，进行页面设置。

Print Setup：弹出对话框，进行打印设置。

Print Preview：当前图形打印预览。

Print：打印当前图形。

2．Edit 菜单

Edit 菜单如图 7-20 所示。

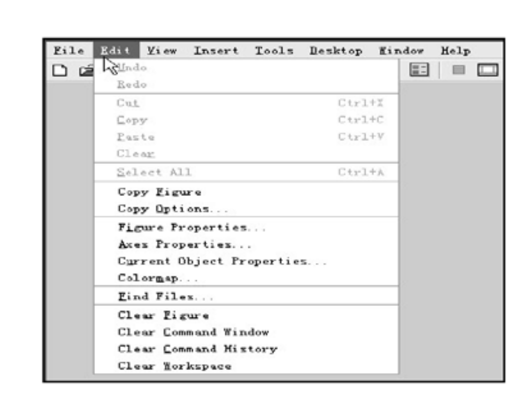


图 7-20　Edit 菜单

Undo：取消上次操作。

Redo：重复上次操作。

Cut：剪切选中图形对象。

Copy：复制选中图形对象。

Paste：粘贴选中图形对象。

Clear：清除选中图形对象。

Select All：选择全部图形对象。

Copy Figure：复制图形窗口内的图形。

Copy Options：复制图形选项。

Figure Properties：浏览图形属性。

Axes Properties：浏览坐标轴对象属性。

Current Object Properties：浏览当前图形对象属性。

Colormap：浏览并修改色图属性。

Find Files：寻找文件中的内容。

Clear Figure：清除图形。

Clear Command Window：清除命令窗口。

Clear Command History：清除命令历史。

Clear Workspace：清除工作区间变量。

3．View 菜单

View 菜单如图 7-21 所示。

Figure Toolbar：确定作图工具栏是否可见。

Camera Toolbar：确定照相工具栏（用于图形观察视角等）是否可见。

Plot Edit Toolbar：确定图形编辑工具栏（用于添加图形对象等）是否可见。

Figure Palette：确定图形面板（用于添加图形对象等）是否可见。

Plot Brower：确定图形对象属性栏是否可见。

Property Editor：确定图形对象属性编辑器是否可见。

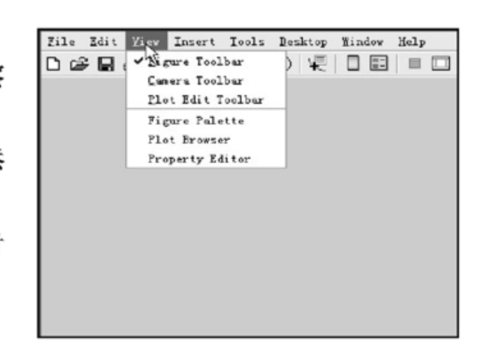


图 7-21 View 菜单

4．Insert 菜单

Insert 菜单如图 7-22 所示。

X Label：插入 *X* 轴标题。

Y Label：插入 *Y* 轴标题。

Z Label：插入 *Z* 轴标题。

Title：插入图形标题。

Lgend：插入图例。

Colorbar：插入颜色刻度。

Line：插入直线。

Arrow：插入箭头。

Text Arrow：插入文本与箭头。

Double Arrow：插入双向箭头。

Text Box：插入文本。

Rectangle：插入矩形。

Ellipse：插入椭圆。

Axes：插入坐标轴。

Light：插入光源。

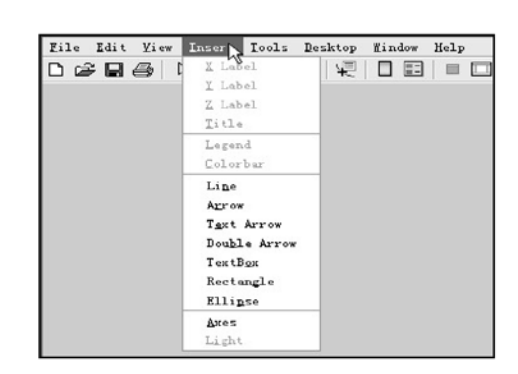


图 7-22 Insert 菜单

5．Tools 菜单

Tools 菜单如图 7-23 所示。

Edit Plot：启用动编辑图形模式。

Zoom In：放大图形。

Zoom Out：缩小图形。

Pan：启用图形移动模式。

Rotate 3D：对图形进行三维旋转。

Data Cursor：启用数据光标模式。

Reset View：视图复位。

Options：缩放与光标设置。

Pin to axes：将图形对象固定到坐标系某一位置。

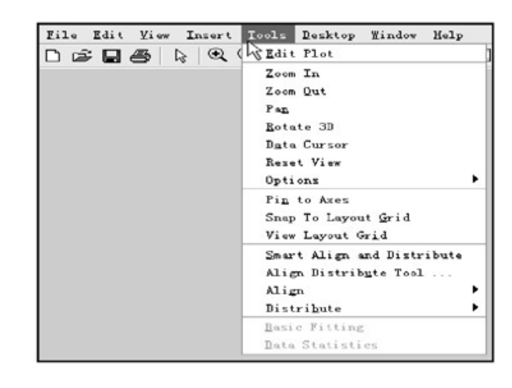


图 7-23 Tool 菜单

Snap To Layout Grid：追使图形对象与网格对齐。

View Layout Grid：显示作图区域网格线。

Smart Align and Dirtribute：把等距离间隔的图形对象按行或列对齐。

Align Distribute Tool：对齐方式与属性（间距等）设置。

Align：对齐方式设置

Distribute：对齐属性（间距等）设置。

Basic Fitting：基本数据拟合。

Data Statistics：显示数据统计特性。

6．Desktop 菜单

Desktop 菜单如图 7-24 所示。

Dock Figure1：将图形 Figure1 锚定到 MATLAB 面板。

7．Windows 菜单

Windows 菜单如图 7-25 所示。

0 MATLAB Command Window：返回 MATLAB 命令窗口界面。

1 Figure No.1：返回图形 1 窗口界面（使用该栏可随时返回不同界面）。

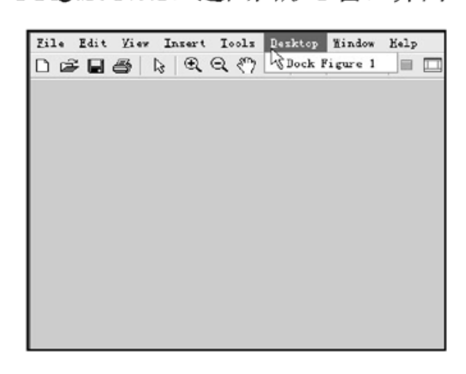


图 7-24 Desktop 菜单



图 7-25 Widow 菜单

8．Help 菜单

Help 菜单如图 7-26 所示。

Graphics Help：进入 MATLAB 自述文件中的图形帮助文件窗口。

Plotting Tools：进入 MATLAB 自述文件中作图工具帮助文件窗口。

Annotating Graphs：进入 MATLAB 自述文件中图形标注帮助文件窗口。

Printing and Exporting：进入 MATLAB 自述文件中打印与输出帮助文件窗口。

Web Resources：使用与 MATLAB 有关的一些 Web 资源。

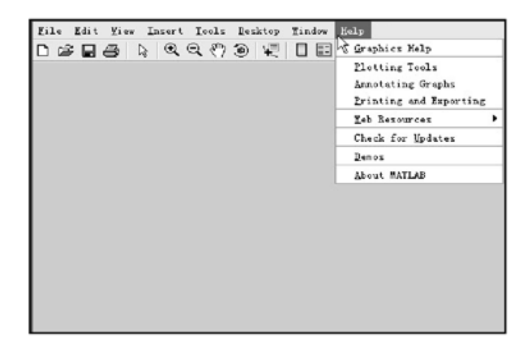


图 7-26 Help 菜单

Check for Update：查询用户 MATLAB 软件有关内容的更新情况。

Demos：进入 MATLAB 自述文件的示例窗口。

About MATLAB：返回关于 MATLAB 的注册信息对话框。

[例 7-18] 利用图形窗口编辑 $u\in[-3,2]$，$v\in[-2,3]$ 上 $u^2-v^2-1=0$ 的图形，将窗口标题改为“绘制函数图形”，不显示系统默认菜单，并适当修改其图形与坐标轴标题。

[解题步骤] 首先使用命令 ezplot('u^2-v^2-1',[-3 2 -2 3])作出图形，然后单击快捷工具栏的按钮（编辑图形），选择图形对象元素进行相应编辑，这里单击图形区域，在弹出的【Property Editor】对话框的【Edit Properties For】中选择“Axes”，将 X 的【Label】属性{u}改为{坐标轴 u}，Y 的 Label 属性{v}改为{坐标轴 v}，Style 的 Title 属性{u}{u}^{2}-{v}^{2}-{1}={0}改为{函数 u^2-v^2-1=0}；在【Edit Properties For】中选择“Figure:1”，将【Menu bar】改为“No menu items(none)”，将【Widow Name】后的编辑框内容变为“绘制函数图形”，【Figure Number】后选择不显示（Show）。

[运行结果] 执行结果见图 7-27。

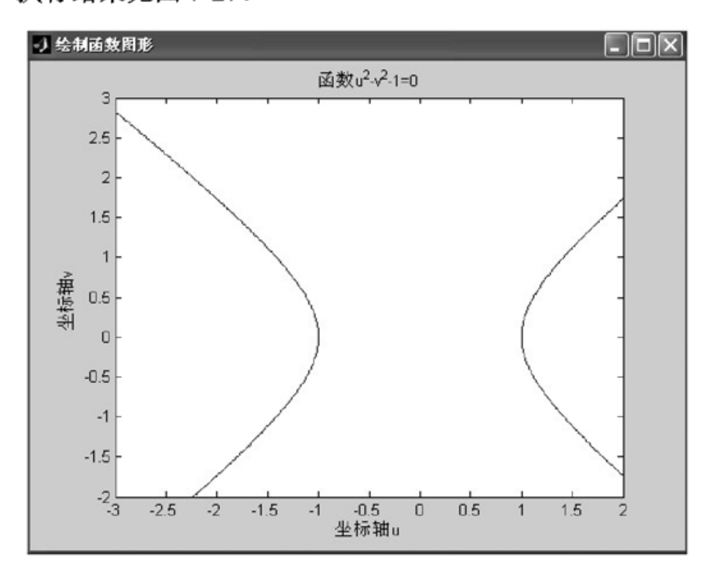


图 7-27 利用图形窗口编辑图形

7.5 声音与动画的实现

7.5.1 声音的实现

[函数命令]

sound

[调用格式]

sound(y),sound(y,f)

sound 为将向量 *y* 传送至扬声器；*f* 为采样频率。

其他的相关命令有：wavplay，wavread，wavercord，wavwrite，分别为*.wav 格式声音的播放、读入、录制、写入，练习时可载入系统提供的 chirp、beep 等声音文件。

[例 7-19] 以 20 000 Hz 制作并播放余弦曲线形式的声音。

[算例代码]

```
%例 7-19
x=cos(linspace(0,10000,20000));              % 制作声音播放向量
sound(x);                                    % 播放声音
```

[运行结果] 若有扬声器，执行算例代码后将听到比较刺耳的声音。

7.5.2 动画的实现

实现动画包括制作动画和播放动画两个过程，分别使用命令 getframe 和 movie。若要在图形界面上对所制作的动画进行动态控制，可以使用滑块控件 slider，通过设置滑块控件的 Value 属性实现用户满意的动态控制。

[函数命令]

getframe

[调用格式]

M=getframe,M=getframe(p),M=getframe(p,r)

getframe 为获得当前图形窗口中的画面；***M*** 为矩阵（用于保存当前图形窗口）；***p*** 为对象句柄；*r* 为图形对象矩形区域，是 4 个元素的向量，r=[left bottom width height]，left、bottom、width、height 分别为矩形区域左下角的纵横坐标、矩形区域的宽度和高度。

[函数命令]

movie

[调用格式]

movie(M,k)

movie 为播放动画；***M*** 为已有的动画帧矩阵；***k*** 为重复播放次数。

[例 7-20] 制作正弦曲线在 $x\in[0,2\pi]$ 中从起点到终点的延伸情况曲线，并播放 3 次。

[算例代码]

```
%例 7-20
s =0.2;x1=0;                       % 确定起始点横坐标 x1 及其增量
nframes = 50;                      % 确定动画总帧数
for k = 1:nframes
      x1= x1 + s;                  % 确定画图时横坐标终止值 x1
      x =0:0.01:x1;
      y =sin(x);
      plot(x,y);                   % 在 x=[0 x1]作 y=sin(x)曲线
      axis([0 2*pi -1 1])          % 定义坐标轴范围
      grid off                     % 不显示网格线
      M(k) = getframe;             % 将当前图形存入矩阵 M(k)
end
movie(M,3)                         % 重复 3 次播放动画 M
```

[运行结果] 得到曲线 y=sin(x)的动画图形。

习题 7

1．试用 MATLAB 的函数文件形式求解下面方程组，并在同一坐标系中画出方程 4 个解

随 a 在区间[0,2]变化的曲线。

$$\begin{cases} ax_1 + x_2 - x_3 = 8 \\ 2x_1 + x_2 - 4x_3 = 5 \\ x_1 + 5x_2 + x_3 = -2 \end{cases}$$

2．已知 x=[1.2 7.0 3.6 5.0 8.0]，y=[4.1 5.2 6.3 9.0 15.0]，z=[11.1 15.2 16.3 19.0 25.0]，试用MATLAB 绘制 x-y 对应的图形与 x–y–z 对应的图形。

3．试用 MATLAB 在同一图形窗口、不同坐标系中分别作出 $y=\cos(x)$、$y=\cos(2x)$、$y=\cos(3x)$、$y=\cos(4x)$在 $x\in[0,2\pi]$ 的图形。

4．试用 MATLAB 在同一直角坐标系中画出函数 $y_1=\sin(x)$、$y_2=\cos(x)$、$y_3=x^2$ 与 $y_4=x$ 在 $x\in[4,10]$内对应的曲线，并标出标题、图例、坐标轴。

5．试用 MATLAB 在同一直角坐标系中画出 A=[0 0.05 1 2 3 4]与 B=[1.521 1.420 1.353 1.212 1.106 0.993]对应的曲线，并标出标题、标题、坐标轴。

6．试用 MATLAB 绘制极坐标系下的图形：$\rho=\cos\left(\dfrac{5\theta}{4}\right)+\dfrac{1}{3}$，其中 $\theta\in[0,8\pi]$。

7．试用 MATLAB 绘制曲线 $y=e^{-0.2x}\sin x$在区间 $x\in[0,5\pi]$ 上的火柴杆图与阶梯图。

8．试用 MATLAB 分别在 $x\in[0,2\pi]$、 $x\in[\pi,3\pi]$ 和 $x\in[-\pi,\pi]$， $y\in[-\pi,\pi]$ 条件下使用命令 fplot 作出函数 $y=x+\cos(x)$ 的图形。

9．试用 MATLAB 在区间 $x\in[0,2\pi]$， $y\in[0,2\pi]$ 作出 $z=x+\sin(y)$ 对应的曲线。

10．试用 MATLAB 在矩形区域 $x\in[-10,10]$， $y\in[-10,10]$ 上分别绘制函数 $z=x^2+y^2$ 与 $y=\dfrac{\sin\sqrt{x^2+y^2}}{\sqrt{x^2+y^2}}$ 对应的三维网格表面图和三维曲面图。

第 8 章　图形用户界面编程

本章要点：

☑ MATLAB 中图形用户界面的创建方法与各组成部分的使用方法；
☑ MATLAB 中图形用户界面编程实现方法；
☑ 图形用户界面编程综合实例，讲述浅基础沉降的可视化编程。

本章讲述 MATLAB 语言中图形用户界面介绍、图形用户界面编程方法、图形用户界面编程实例等几方面的内容。

8.1　图形用户界面的创建与组成

8.1.1　创建图形用户界面

图形用户界面，简称 GUI，是英文 Graphic User Interface 的缩写。其创建方法是：使用菜单【File】→【New】→【GUI】或者在命令窗口使用命令 guide，弹出如图 8-1 所示的对话框，在该对话框内可以创建图形用户界面（Create New GUI）或者通过【Browse】打开一个已有图形用户界面（Open Existing GUI），如图 8-1 所示，在创建新的图形用户界面时，可以进行适当设置，图 8-1 中的 Blank（默认），GUI with Uicontrols，GUI with Axes and Menu，Modal Question Dialog 分别为图形用户界面的默认设置（图内内容为空）、含有控件、含有坐标系、含有问题对话框。在选择了一个已有图形界面或设置完成后，单击【OK】，将得到已编界面或新界面，新创建的默认图形用户界面如图 8-2 所示。

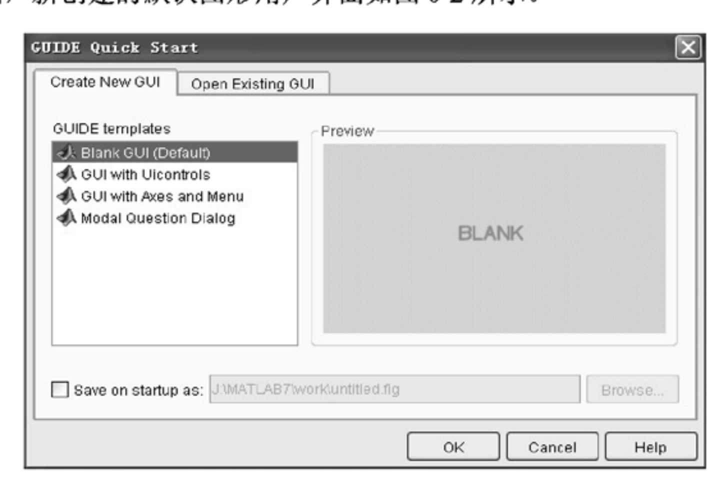


图 8-1　创建图形用户界面

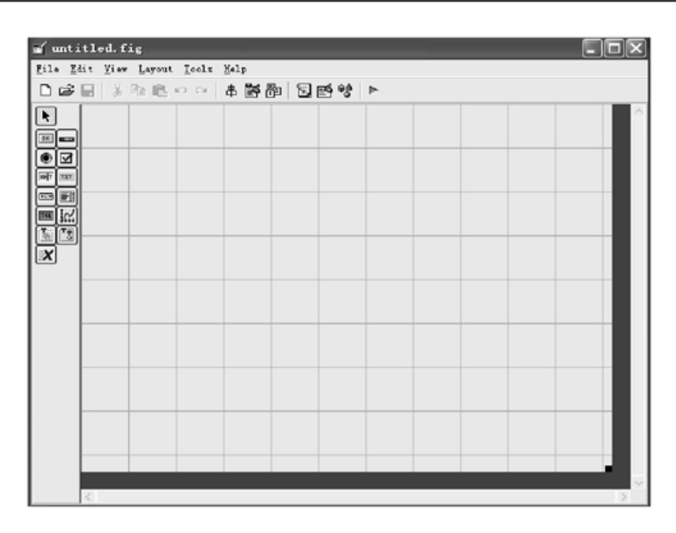


图 8-2 新建的空白图形用户界面

8.1.2 图形用户界面介绍

图形用户界面由下述几部分组成：布局区域（Layout Area），对象对齐工具（Align Object），菜单编辑器（Menu Editor），切换顺序编辑器（Tab Order Editor），m 文件编辑器（M-file Editor），属性检查器（Property Inspector），图形对象浏览器（Object Brower），激活按钮（Run），取消和重复上次操作按钮（Undo 和 Redo），组件面板（Component Pallette）。下面分别介绍。

1．布局区域（LayoutArea）

布局区域（Layout Area）是用户在图形用户界面进行操作的的主要区域，用于 MATLAB 下图形操作的外观设置，一旦被激活，其就成为图形窗口。

在布局区域内添加控件时，可在组件面板（Component Pallette）中单击所选控件（光标成为十字形），然后在布局区域中单击某点，该点即为控件的左上角位置；也可先单击所选控件，然后拖动光标并设置控件的大小。

2．对象对齐工具（Align Object）

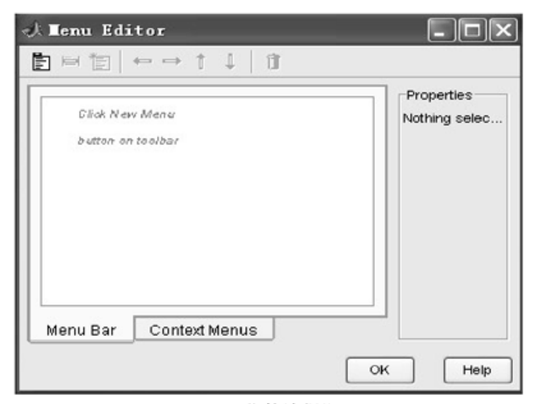
单击对象对齐工具（Align Object），显示对齐工具按钮，如图 8-3 所示，通过使用 Ctrl 键选择多个对象，再通过如图 8-3 所示的相关设置来调整所选择对象之间的相对位置和间距，单击【OK】完成，若单击【Apply】则当前对象调整完成并可继续对新对象进行对齐设置。



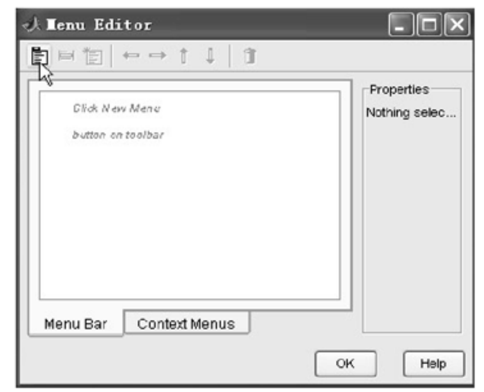
图 8-3 对象对齐工具（Align Object）

3．菜单编辑器（Menu Editor）

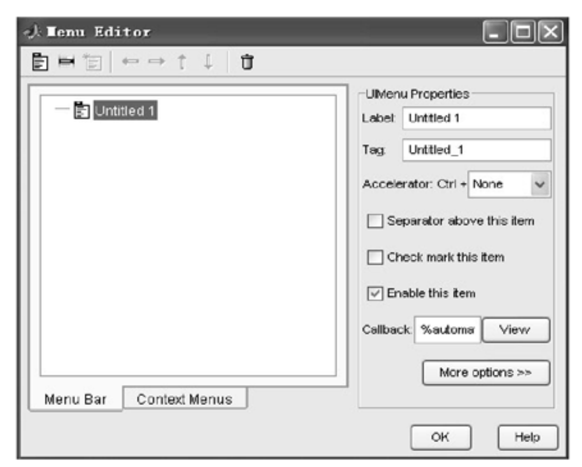
单击菜单编辑器（Menu Editor）或布局区域单击鼠标右键并选择【Menu Editor】，弹出如图 8-4（a）所示的菜单编辑器对话框，可以在该对话框中进行菜单条对象（显示为图形窗口菜单）和上下文菜单（用户右击图形对象弹出菜单）的创建与设置，分别如图 8-4（b）和图 8-4（c）所示。



（a）菜单编辑器



（b）创建菜单



（c）设置菜单

图 8-4　菜单编辑器、创建菜单与设置菜单

4．切换顺序编辑器（Tab Order Editor）

单击切换顺序编辑器（Tab Order Editor），弹出切换顺序编辑器，如图 8-5 所示，可进行图形对象的快速切换。

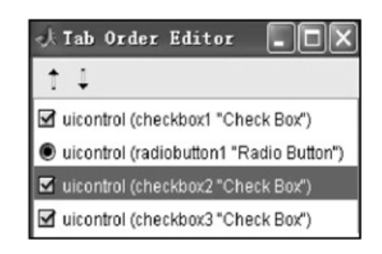


图 8-5　切换顺序编辑器（Tab Order Editor）

5．文件编辑器（M-file Editor）

单击 m 文件编辑器（M-file Editor），或布局区域选择某个图形组件单击鼠标右键并选择【Menu Editor】，可以进行有关 m 文件的编辑，对应的回调函数（callback）是图形对象被选中时执行的一段代码，其代码体一般包括：获得触发动作对象的句柄、获得图形元素的属性、获得目标对象的句柄、设置目标对象的属性。

6．属性检查器（Property Inspector）

在布局区域选择某个图形组件，再单击属性检查器工具条按钮，或单击鼠标右键并选择【Property Inspector】，如图 8-6 所示，弹出类似于图 8-7 所示的属性检查器对话框，它用于检查和修改图形对象的某些属性。

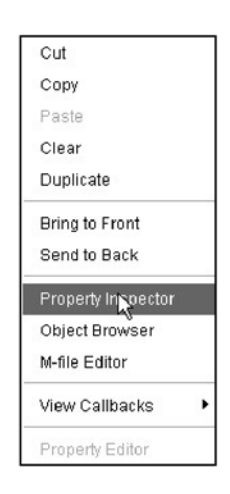


图 8-6 进入属性检查器

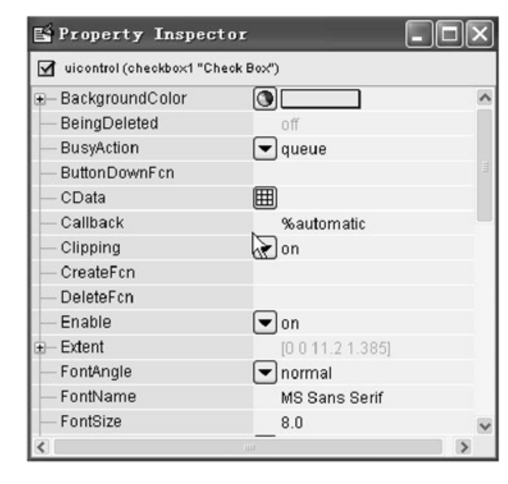


图 8-7 属性检查器

7．图形对象浏览器（ObjectBrower）

在布局区域选择某个图形组件，再单击图形对象浏览器（Object Brower），或单击鼠标右键并选择【Object Brower】，弹出图形对象浏览器对话框显示图形界面中各个对象的层次关系，如图 8-8 所示。

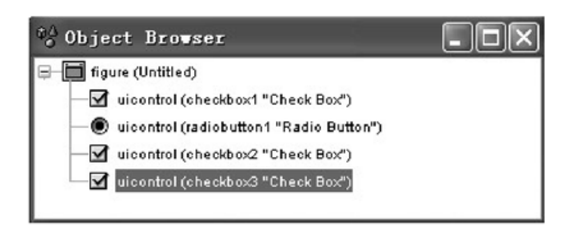


图 8-8 图形对象浏览器

8．激活按钮（Run）

激活按钮（Run Button）用于图形执行，如果是新建图形用户界面，则 MATLAB 将弹出保存【Save As】对话框，如图 8-9 所示。一般情况下，单击激活按钮（Run），相当于其他程序设语言的编译、连接与执行。

9．取消和重复上次操作按钮（Undo 和 Redo）

按钮（Undo）和（Redo），用于图形用户界面上次操作的取消和重复。

10．组件面板（ComponentPallette）

为了方便可视化编程，MATLAB 的组件面板（Component Pallette）提供了较多的控件，如图 8-10 所示。由于 MATLAB 是英文版，为用户使用方便，表 8-1 列出了各控件的中英文含义。

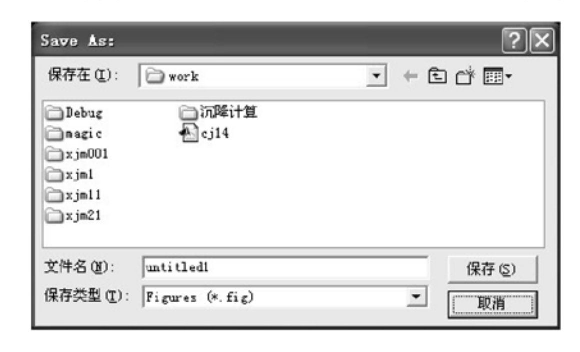


图 8-9　保存对话框



图 8-10　组件面板（Component Pallette）

表 8-1　组件面板的控件名称

控 件 图 形	控件英文名称	控件中文名称
	Push Button	鼠标按下操作按钮
	Radio Button	单项选择按钮
	Edit Text	编辑框
	Pop-up Menu	组合框
	Toggle Button	切换（按下/弹起）按钮
	Panel	面板控件
	ActiveX Control	Activex 控制控件
	Slider	滑动条
	Check Box	多项选择
	Static Text	静态文本
	Listbox	列表框
	Axes	坐标系
	Button Group	按钮组

8.2　图形用户界面编程基础

MATLAB 中使用图形用户界面进行编程时，经常会碰到不同的图形对象，这些对象主要包括窗口对象、菜单对象、对话框对象、控件对象和坐标轴对象。下面分别介绍各对象有关属性的设置方法。

8.2.1　窗口对象

1．窗口对象属性的可视化设计

窗口的基本属性包括位置（Posiotion）、窗口编号（Number Title）、标题栏（Name）和

菜单（Menubar）。分别简单介绍如下。

Position 属性：设置窗口位置，它为 1×4 的向量，前后两个数值分别为窗口矩形左下角及右上角的横坐标与纵坐标数值。

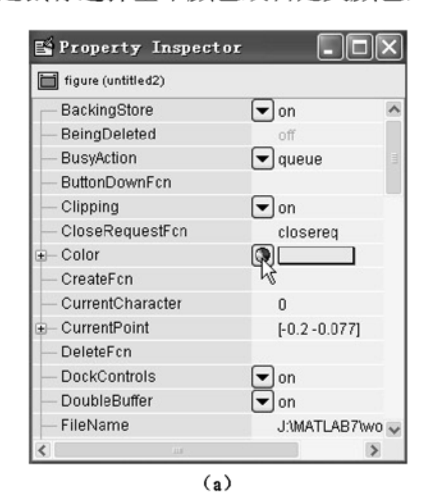
Name 属性：窗口标题栏对应的字符串。

NumberTitle 属性：窗口编号选项，有两个：选项 on（默认选项）显示窗口编号；选项 off 则不显示窗口编号。

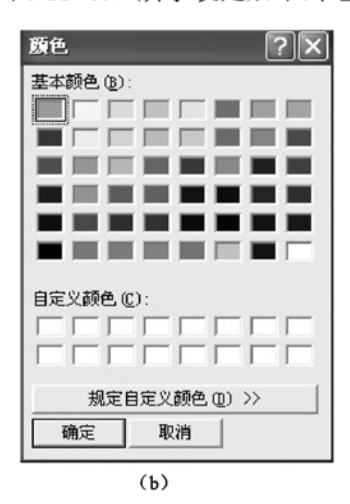
Menubar 属性：菜单栏目选项，有两个：选项 Figure 为选择系统默认菜单；选项 None 则不选用系统默认菜单。

窗口还有颜色（Color）、调整（Resize）和可见性（Visible）等属性。

Color 属性：在窗口任一位置单击右键或打开属性检查器（Property Inspector），如图 8-11（a）所示，在【Property Inspector】窗口中单击 Color 设置按钮，再在弹出的【颜色】对话框内通过鼠标选择基本颜色或自定义颜色，如图 8-11（b）所示设定后单击【确定】按钮。



（a）



（b）

图 8-11　窗口颜色的可视化设置

Resize 属性：设置窗口大小是否可调整，有两个选项：选项 on（默认选项）为可调整；选项 off 则不可调整。

Visible 属性：设置窗口是否可见，有两个选项：选项 on（默认选项）为可见；选项 off 则不可见。

2．窗口对象属性程序设计

创建方法：

set(对象句柄,属性 1,属性值 1,属性 2,属性值 2,…)

获取方法：

get(对象句柄,属性 1,属性值 1,属性 2,属性值 2,…)

[例 8-1]　试在位置[200,300,600,400]创建含有系统默认菜单，显示标题“可视化编程”但不显示窗口编号的图形窗口。

[算例代码]

```
%例 8-1
clear;
```

```
h=figure;
set(h,'Position',[100 150 500 400],'Menubar','figure',...
        'Name','可视化编程','NumberTitle','off' )
```

[运行结果]

运行结果如图 8-12 所示。

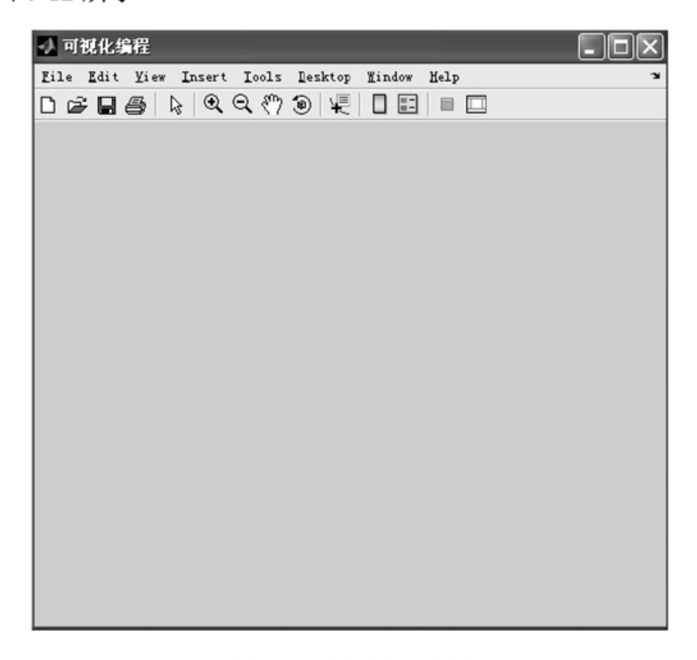


图 8-12　图形窗口设置

8.2.2　菜单对象

1．菜单对象属性的可视化设计

通过属性检查器进行菜单可视化设计时，有两个可以切换的选项，即 MenuBar 和 ContextMenu，分别创建下拉式菜单和上下文菜单。对于下拉式菜单来说，如果窗口的 Menubar 属性设置为“figure”、则有系统默认的 8 个原始菜单：File、Edit、View、Insert、Tools、Desktop、Window 和 Help。如果用户要调整各菜单位置或添加新的菜单，可直接使用菜单编辑器。如果窗口的 Menubar 属性设置为“none”，则不显示 8 个原始菜单，但可以添加新的菜单。用户创建好菜单后，执行相应的操作可以通过在菜单选项中选择相应的回调函数 Callback。

重要提示：Callback 函数对应的 m 文件必须处于工作目录下。

2．菜单对象属性程序设计

创建方法：

```
uimenu(窗口句柄,属性 1,属性值 1,属性 2,属性值 2,…)
```

窗口句柄如果使用“gcf”，是指当前图形窗口（意即 Get Current Figure）。对于当前图形窗口来说，“窗口句柄”一项可以省略。

[例 8-2]　在创建图形窗口的基础上，创建名为“输入一般资料”的菜单，同时创建其下级菜单“输入基础资料”。

[算例代码]

```
%例 8-2
clear;
h=figure;
set(h,'Position',[100 150 500 400],'Menubar','figure',...
        'Name','可视化编程','NumberTitle','off')
h1=uimenu(h,'Label','输入一般资料');
h11=uimenu(h1,'Label','输入基础资料','Callback','aaa1.m');
```

[运行结果]

运行结果如图 8-13 所示。

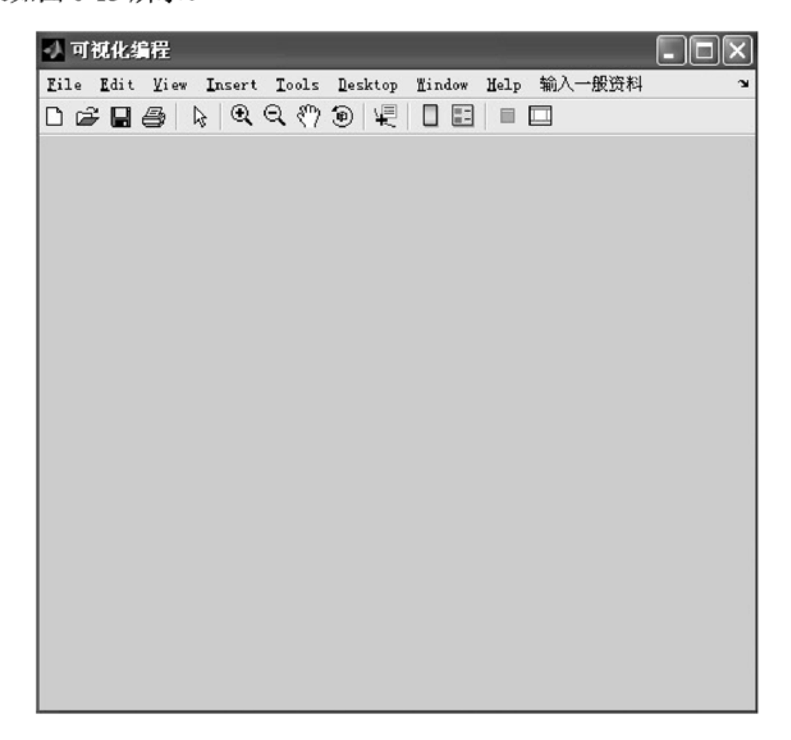


图 8-13　创建图形窗口与菜单

8.2.3　对话框对象

对话框对象包括错误提示对话框、帮助提示对话框和信息提示对话框，调用格式分别为 errordlg('s1','name')、helpdlg('s1','name')和 messagebox('s1','name')，s1 为对话框内的字符串，name 为对话框标题。

[例 8-3] 生成标题为“基本信息”、提示内容为“这里给出的是基本信息”的错误提示对话框。

[算例代码]

```
%例 8-3
errordlg('这里给出的是基本信息','基本信息');
```

[运行结果]

运行结果如图 8-14 所示。



图 8-14　错误提示对话框的创建

8.2.4　控件对象

1．控件对象属性的可视化设计

对于表 8-1 的各基本控件，选中使其成为十字形，直接在布局区域内单击即可进行布置。而要对布局区域内的各控件进行属性设置，可以单击对应控件，然后单击鼠标右键，通过属性检查器（Preoperty Inspector）修改其属性值。

重要提示：Callback 函数是最重要的属性选项，对应的 m 文件必须处于工作目录下；整个程序设计过程，通常主要是 Callback 函数的编辑。

各控件的常用属性值见表 8-2。

表 8-2　控件的基本属性

控件属性名称	控件属性值使用说明
Style	控件类型，见表 8-1
Position	控件相对于窗口左下角位置及其宽度与高度
BackgroundColor	设置背景颜色
ForegroundColor	设置前景颜色
String	显示在控件上的字符串
Tag	控件标签名称，图形对象句柄通常通过这一属性值调用与赋值
Callback	回调函数（通常是 m 文件，也可为命令字符串）

2．窗口对象属性程序设计

创建方法：

uicontrol(窗口句柄,属性 1,属性值 1,属性 2,属性值 2,…)

窗口句柄如果是当前图形窗口，“窗口句柄”一项可省略。

[例 8-4]　在创建图形窗口的基础上，在适当位置创建一个“基础长度”静态文本框。

[算例代码]

```
%例 8-4
clear;
h=figure;
set(h,'Position',[100 150 500 400],'Menubar','figure',...
    ' Name','可视化编程','NumberTitle','off')
h1=uicontrol(h,'Style','text','Position',[100 150 80 20],'String','基础长度');
```

[运行结果]

运行结果如图 8-15 所示。



图 8-15 在图形窗口内创建控件

8.2.5 坐标轴对象属性

1．坐标轴对象属性的可视化设计

选中表 8-1 中控件使其成为十字形，在布局区域内单击即可。要对布局区域内的坐标轴对象设置属性，可以单击对应坐标轴对象并单击右键，通过属性检查器（Preoperty Inspector）修改其属性值。坐标轴对象的常用属性值见表 8-3。

表 8-3 坐标轴对象的常用属性

属 性 名 称	属性值使用说明
Xdir,Ydir,Zdir	*X*、*Y*、*Z*轴的方向
Xdim,Ydim,Zdim	*X*、*Y*、*Z*轴的数据范围
XColor,Ycolor,ZColor	*X*、*Y*、*Z*轴的颜色
XGrid,YGrid,ZGrid	*X*、*Y*、*Z*轴的网格线（on 为显示网格，off 为不显示，默认为 off）
FileName,Name	图形文件位置与名称
MenuBar	有无菜单（figure 为系统默认菜单，none 则不显示菜单）

2．窗口对象属性程序设计

创建方法：

axes(窗口句柄,属性 1,属性值 1,属性 2,属性值 2,…)

“窗口句柄”一项在运行窗口是当前图形窗口时可省略。

[例 8-5] 在创建图形窗口的基础上，在适当位置创建一个坐标轴对象并在该坐标系内作出函数 y=sin(x)的图形。

[算例代码]

```
%例 8-5
clear;
h=figure;
set(h,'Position',[100 150 500 400],'Menubar','figure',...
        'Name','可视化编程','NumberTitle','off')
axes('position',[0.1 0.7 0.8 0.2])
x=0:0.01:2*pi;
y=sin(x);
plot(x,y);
```

[运行结果]

运行结果如图 8-16 所示。

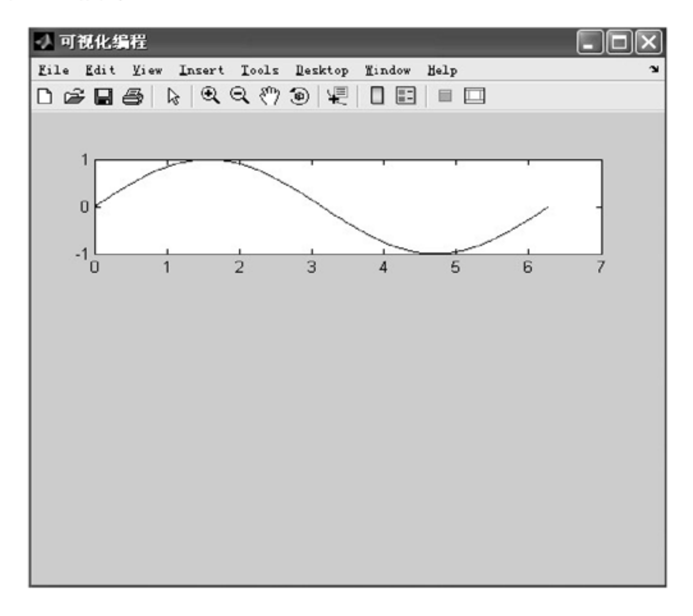


图 8-16　在图形窗口内创建坐标轴对象并绘制图形

8.2.6　图形用户界面编程过程

图形用户界面编程可采用可视化编程方式或代码直接编程方式。如果涉及控件的操作控制，由于常常涉及函数计算或数据处理，采用代码直接编程则更为方便一些。图形界面的外观设计一般采用可视化编程方式，其包括如下过程：

（1）进入 GUI（图形用户界面）；

（2）添加图形对象；

（3）修改菜单属性；

（4）修改图形对象属性；

（5）编辑回调函数或命令；

（6）另存为图形文件；

（7）打开图形文件，对有关对象属性进行修改。

对于实际编程问题来说，可以采用两者结合的方式。无论采取哪一种编程方式，通常都要进行回调函数（m 文件）的编制。就编制回调函数来说，经常要使用三个函数命令：

findobj，set 和 get。命令 findobj 用于找出图形对象内具有某一标签属性的具体对象；命令 set 用于具体对象的属性及其属性值的设置；命令 get 用于具体对象的属性及其属性值的获得。关于命令 set 和 get 的用法，前面已经讲到，不再赘述。下面介绍命令 findobj 的使用方法。

[使用格式]

H=findobj(图形对象句柄,属性,属性值)

H 为返回句柄，图形对象句柄可以是 gcbf、gcbo、gco、gcf、gca 之一，含义分别为获得具有返回控制的当前图形句柄、获得具有返回控制的当前对象句柄、获得当前对象句柄、获得当前图形句柄、获得当前坐标轴对象句柄。

下面以浅基础沉降计算的综合实例，说明图形用户界面编程在实际工程的应用。

8.3 图形用户界面编程综合实例：浅基础沉降计算

8.3.1 理论基础

如图 8-17 所示，设基础底面以下土层共有 m 层，第 j（j=1,…,m）层分为 n 层，第 i 细层（i=1,…,n）厚度 $h_{(j,i)}$，该细层在自重应力 $p_{1(j,i)}$ 与附加应力 $p_{2(j,i)}$ 作用下，对应的孔隙比由 $e_{1(j,i)}$ 变为 $e_{2(j,i)}$，基础底面的最终沉降量为 $S=\sum_{j=1}^{m}\sum_{i=1}^{n}\frac{e_{1(j,i)}-e_{2(j,i)}}{1+e_{1(j,i)}}h_{(j,i)}$ 。

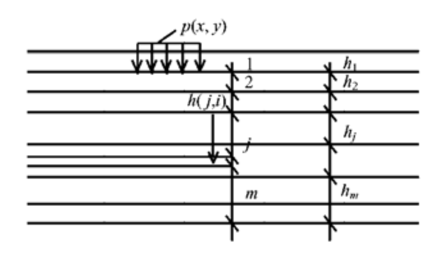


图 8-17 沉降计算剖面

要计算浅基础沉降，必须计算各细层的自重应力与附加应力，还要确定自重应力与总应力（自重应力和附加应力之和）对应的孔隙比。自重应力 $p_{(j,i)}$ 可以根据土层埋深（或厚度）及土的基本物理力学性质指标确定。附加应力由 Boussineq 解或 Mindlin 解积分获得。各向同性半无限空间均质弹性体在表面一点（坐标为（0,0,0））与地面以下一点（坐标（0,0,c））的垂直集中力 Q 作用下，表面以下任意点（设其坐标为（x,y,z））的竖向附加应力为（分别为 Boussineq 解与 Mindlin 解）：

$$\sigma_z=\frac{3Qz^3}{2\pi[x^2+y^2+z^2]^{5/2}}$$

$$\sigma_z=\frac{Q}{8\pi(1-\mu)}\left\{\frac{(1-2\mu)(z-c)}{R_1^3}-\frac{(1-2\mu)(z-c)}{R_2^3}+\frac{3(3-4\mu)z(z+c)^2-3c(z+c)(5z-c}{R_2^5}\right.$$
$$\left.+\frac{3(z-c)^3}{R_1^5}+\frac{30cz(z+c)^3}{R_2^7}\right\}$$

式中，μ 为土的泊松比，$R_1=\sqrt{x^2+y^2+(z-c)^2}$，$R_2=\sqrt{x^2+y^2+(z+c)^2}$。基础底面 F 范围内作用分布荷载 $p(x,y)$时，土中某点 $M(x,y,z)$ 的竖向附加应力为 $\sigma_z=\iint_F \mathrm{d}\sigma_z=\iint_F \sigma(x,y,z,\mu,c,\xi,\eta)p(x,y)\mathrm{d}\xi\mathrm{d}\eta$。

自重应力与总应力对应的孔隙比，可通过压缩试验成果确定。根据室内土工压缩试验，可得应力$\{p_{(i,k)}\}$与对应孔隙比$\{e_{(i,k)}\}$（k=1,…,l,l为加荷级数）及相应的压缩曲线（e–p 曲线），压缩曲线可假定为双曲线形式 $e=e_0+p/(a+bp)$，拟合参数 a, b 可用非线性最小二乘法命令 lsqnonlin 确定。

进行图形用户界面设计时，可将压缩层计算深度、土层分层数直接在界面上输入，通过不同控件的回调函数计算不同深度的附加应力与基础底面的沉降。

8.3.2　编程实现

1．进入 MATLAB 程序设计平台

选择【开始】→【程序】→【MATLAB 7.0】→【MATLAB 7.0】，进入 MATLAB 7.0 程序设计平台。

2．进行图形用户界面外观设计

在 MATLAB 7.0 程序设计平台中，选择【File】→【New】→【GUI】，在弹出对话框中选择【BlankGUI(Default)】，显示图形用户设计平台。在该平台中进行外观设置，菜单及有关属性设置见表 8-4、表 8-5 和表 8-6，对应的外观如图 8-18 所示，同时，将图保存至工作目录。

表 8-4　沉降计算程序外观设置_窗口设置

属 性 名 称	窗　口
FileName	<MATLAB7>\work\cj14.fig
Menubar	None
NumberTitle	Off
Name	浅基础沉降计算
Position	[10 26 138 37]

表 8-5　沉降计算程序外观设置_菜单设置

属 性 类 型	Label	CallBack
菜　单	文件	filemenufcn(gcbf,'FilePost')
	打开	filemenufcn(gcbf,'FileOpen')
	关闭	close(gcbf)
	保存	filemenufcn(gcbf,'FileSave')
	另存为	filemenufcn(gcbf,'FileSaveAs')
	打印	printdlg
	编辑	editmenufcn(gcbf,'EditPost')
	复制图形	editmenufcn(gcbf,'EditCopyFigure')
	工具	domymenu('menubar','updatetools',gcbf)
	显示图形工具	domymenu('menubar','toggletoolbar',gcbf)
	图面编辑	plotedit(gcbf)

表 8-6 沉降计算程序外观设置_控件设置

属 性 名 称	Callback	String	Style	Tag
控 件		基础与荷载条件	listbox	listbox1
		荷载情况数	text	text11
		1	edit	edit101
	aaa1	确认	pushbutton	pushbutton1
	cjb1.m	确认	pushbutton	pushbutton2
		土层条件	listbox	listbox2
		土层数	text	text21
		2	edit	edit102
		地下水埋深	text	text22
		0.5	edit	edit103
	aaa2.m	确认	pushbutton	pushbutton3
	cjb2.m	确认	pushbutton	pushbutton4
		应力沉降计算	listbox	listbox1
	cjb3.m	e–p曲线拟合结果	pushbutton	pushbutton5
	aaa41.m	该点沉降计算值:	pushbutton	pushbutton6
	aaa42.m	绘制应力分布图形	pushbutton	pushbutton7

特别提示：注意输入法，撇号应该使用“'”、而不用“’”；若回调函数使用当前目录下的 aaa1.m，则其 Callback 属性应使用 aaa1。

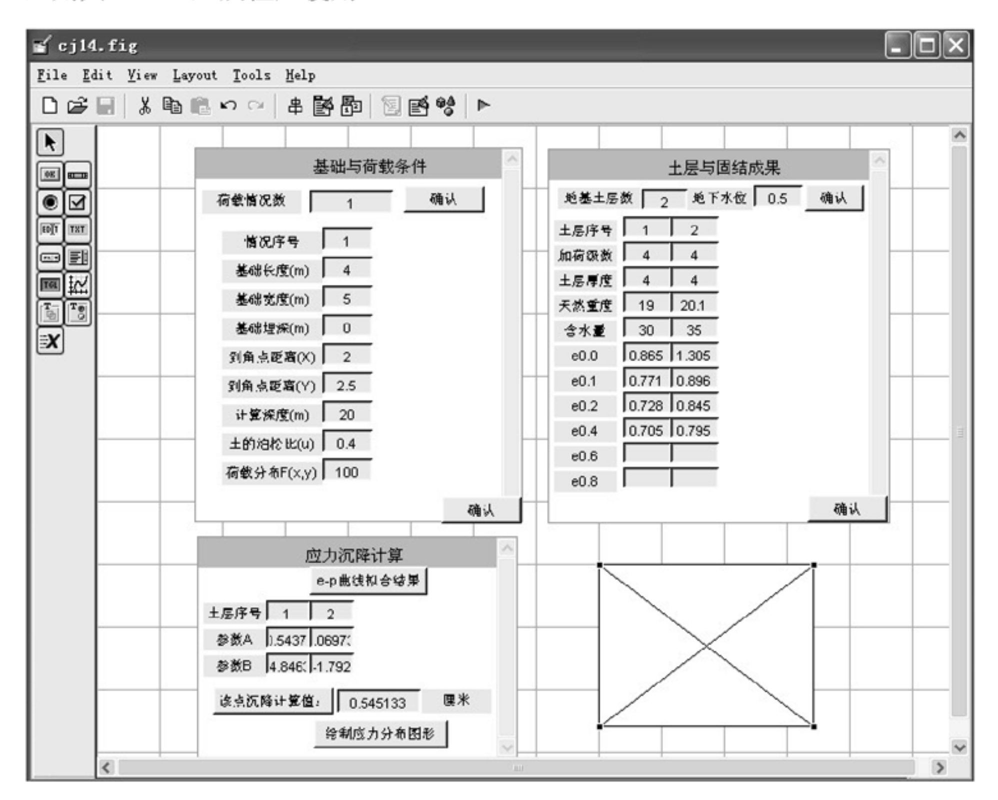


图 8-18 沉降计算程序外观

3．输入菜单与控件的回调函数。

在工作目录下，编辑各回调函数代码，完整代码如下。

aaa1.m 代码如下：

```
clear
% 输入基础参数
qw1=findobj(gcbf,'tag','edit101');q1=get(qw1,'string');n1=eval(q1);
e(1)=uicontrol('Style', 'text','string','情况序号');
e(2)=uicontrol('Style', 'text','string','基础长度(m)');
e(3)=uicontrol('Style', 'text','string','基础宽度(m)');
e(4)=uicontrol('Style', 'text','string','基础埋深(m)');
e(5)=uicontrol('Style', 'text','string','到角点距离(X)');
e(6)=uicontrol('Style', 'text','string','到角点距离(Y)');
e(7)=uicontrol('Style', 'text','string','计算深度(m)');
e(8)=uicontrol('Style', 'text','string','土的泊松比(u)');
e(9)=uicontrol('Style', 'text','string','荷载分布 F(x,y)');
for i=1:9,
    set(e(i),'Position', [ 70    410-22*(i-1) 80 18]);
end
for i=1:9
    for j=1:n1,
        a1(i,j)=uicontrol('Style', 'edit','Position', [150+(j-1)*40    410-22*(i-1) 40 18]);
    end
end
for j=1:n1,
    set(a1(1,j),'string',j);
end
% 输入基础参数初始值
for j=1:n1,
    set(a1(2,j),'string',4);
end
for j=1:n1,
    set(a1(3,j),'string',5);
end
for j=1:n1,
    set(a1(4,j),'string',0);
end
for j=1:n1,
    set(a1(5,j),'string',2);
end
for j=1:n1,
    set(a1(6,j),'string',8);
end
for j=1:n1,
    set(a1(7,j),'string',20);
end
for j=1:n1,
    set(a1(8,j),'string',0.4);
```

```
end
for j=1:n1,
    set(a1(9,j),'string',100);
end
```

cjb1.m 代码如下：

```
% 获得基础参数
qr=get(a1,'string');qw=char(qr);qe=str2num(qw);
for i=1:n1,
    qq(1,i)=0;
end,
qq(1,1)=n1;
for i=1:n1,
    h{i}=char(get(a1(9,i),'string'));
end
for j=1:8,
    for i=1:n1,
        qa1(j,i)=eval(get(a1(j,i),'string'));
    end;
end;
w1=[qq;qa1];
```

aaa2.m 代码如下：

```
% 输入地基参数
qw21=findobj(gcbf,'tag','edit102');
q21=get(qw21,'string');
n2=eval(q21);
qw22=findobj(gcbf,'tag','edit103');
q22=get(qw22,'string');
d22=eval(q22);
e(1)=uicontrol('Style', 'text','string','土层序号');
e(2)=uicontrol('Style', 'text','string','加荷级数');
e(3)=uicontrol('Style', 'text','string','土层厚度');
e(4)=uicontrol('Style', 'text','string','天然重度');
e(5)=uicontrol('Style', 'text','string','含水量');
e(6)=uicontrol('Style', 'text','string','e0.0');
e(7)=uicontrol('Style', 'text','string','e0.1');
e(8)=uicontrol('Style', 'text','string','e0.2');
e(9)=uicontrol('Style', 'text','string','e0.4');
e(10)=uicontrol('Style', 'text','string','e0.6');
e(11)=uicontrol('Style', 'text','string','e0.8');
for i=1:11,set(e(i),'Position', [ 365    410-20*(i-1) 50 17]);end
for j=1:n2
    for i=1:11,
        a2(i,j)=uicontrol('Style', 'edit','Position', [420+(j-1)*38    410-20*(i-1) 38 17]);
    end
end
```

```
for j=1:n2,
    set(a2(1,j),'string',j);
end
for j=1:n2,
    set(a2(2,j),'string',4);
end
for j=1:n2,
    set(a2(3,j),'string',4);
end
for j=1:n2,
    set(a2(4,j),'string',19.0);
end
for j=1:n2,
    set(a2(5,j),'string',30.0);
end
for j=1:n2,
    set(a2(6,j),'string',0.865);
end
for j=1:n2,
    set(a2(7,j),'string',0.771);
end
for j=1:n2,
    set(a2(8,j),'string',0.728);
end
for j=1:n2,
    set(a2(9,j),'string',0.705);
end
```

cjb2.m 代码如下：

```
% 获得基础参数
for i=1:12,
    for j=1:n2,
        w2(i,j)=0.0;
    end;
end;
w2(1,1)=n2;
w2(1,2)=d22;
for i=1:n2,
    w2(2,i)=i;
end;
for i=1:n2,
    ht(i)=0.0;
end
for i=1:n2,
    ht(i)=eval(get(a2(2,i),'string'));
end
for i=1:n2,
    w2(3,i)=ht(i);
```

```
end
for i=1:n2,
    for j=4:w2(3,i)+6,
        w2(j,i)=eval(get(a2(j-1,i),'string'));
    end;
end;
```

cjb3.m 代码如下：

```
% 土层参数拟合
qw2=findobj(gcbf,'tag','edit102');
q2=get(qw2,'string');
n=eval(q2);
f1(1)=uicontrol('Style', 'text','string','土层序号');
f1(2)=uicontrol('Style', 'text','string','参数 A');
f1(3)=uicontrol('Style', 'text','string','参数 B');
for i=1:3,
    set(f1(i),'Position',[65   143-20*i 50 15]);
end
for i=1:3
    for j=1:n,
        g1(i,j)=uicontrol('Style', 'edit','Position', [100+j*35   145-20*i 50 15]);
    end
end
for j=1:n,
    set(g1(1,j),'string',j);
end
%双曲线法计算土的压缩特性
for m=1:n2;
    w3(m,1)=0.0;
    w3(m,2)=0.000;
end;
for m=1:n2;
    for k=1:w2(3,m),
        pu(k)=0.0;
        eu(k)=0.0;
    end;
    for k=1:w2(3,m),
        eu(k)=w2(6+k,m);
        if k==1,
            pu(k)=0.0;
        end;
        if k==2,
            pu(k)=0.1;
        end;
        if k==3,
        pu(k)=0.2;
    end;
    if k==4,
```

```
                pu(k)=0.4;
            end;
            if k==5,
                pu(k)=0.6;
            end;
            if k==6,
                pu(k)=0.8;
            end;
        end
        for i=1:w2(3,m)-1,
            xf(i)=pu(i+1)-pu(1);
            yf(i)=eu(i+1)-eu(1);yf(i)=xf(i)/yf(i);
        end;
        fun=inline('x(1)+x(2)*xf','x','xf');
        x=lsqcurvefit(fun,[1 2],xf,yf);
        ws1=num2str(x(1));
        ws2=num2str(x(2));
        w3(1,m)=x(1);
        w3(2,m)=x(2);
        set(g1(2,m),'string',ws1);
        set(g1(3,m),'string',ws2);
    end
```

aaa41.m 代码如下：

```
    % 计算沉降
      % 1.确定基础资料//基础长度 a(i)，基础宽度 b(i)，基础埋深 h0(设为一个埋深)
    for i=1:n1,
        a(i)=w1(3,i);
    end;
    for i=1:n1,
        b(i)=w1(4,i);
    end;
    h0=w1(5,1);
      % 2.确定土层情况与计算深度//泊松比 u（设为一个）、厚度 H(i)(设第一层土厚度比基础埋深大)、
      %     重度 gama(i)、深度 HH(i)、土层拟合参数 q1(i)与 q2(i)
      %     计算深度 H1（设为最后一个土层的埋深）
    u=w1(9,1);
    H1=w2(4,n2);
    H1=w2(4,n2);
    for i=1:n2;
        t1(i)=0;
        t2(i)=0;
        H(i)=0.0;
    end;
    for i=1:n2,
        t1(i)=w3(1,i);
    end;
    for i=1:n2,
```

```
        t2(i)=w3(2,i);
end;
for i=1:n2,
        H(i)=w2(4,i);
end;
H(1)=H(1)-h0;
H=[h0 H];
hh=cumsum(H);
for i=1:n2,
        e0=w2(7,i);
        ga=w2(5,i);
        w0=w2(6,i)*0.01;
        ds=ga/10*(1+e0)/(1+w0);
        r(i)=(ds-1)/(1+e0)*10;
end
   % 3.计算各层面(基础底面、第一层下层面、第二层下层面)处自重应力
for i=1:n2+1,
        c(i)=0.0;
end;
c(1)=r(1)*h0;
for i=2:n2+1,
        c(i)=c(i-1)+r(i-1)*H(i);
end;
   % 4.计算各层土的变形
        %对深度、自重应力与总应力的整体矩阵赋初值
for i=1:20*n2,
        tq(i)=0;
        tq2(i)=0;
        tq3(i)=0;
end;
for k=1:n2
        % 4.1.计算各点自重应力
        for j=1:20,
                hhz(j)=0.0;
                hz(j)=j*H(k+1)/20;
                sig1(j)=c(k)+r(k)*hz(j);
                tq2(j+20*(k-1))=sig1(j);
        end
        % 4.2.计算各层附加应力
        for i=1:n1,
                wq(i)=0.0;
        end;
        for j=1:20;
                z=hh(k)+j*H(k+1)/20-h0;
                ww(j)=0.0;
                tq1(j+20*(k-1))=z+h0;
                for i=1:n1;
                        jss=char(h(i));
```

```
                x0=w1(3,i);
                y0=w1(4,i);
                x1=w1(6,i);
                y1=w1(7,i);
                syms x y t1 t2 t3 t4 t5 t6 t7;
                t1=3;
                t2=2;
                t3=pi;
                t4=z*z*z;
                t5=(x*x+y*y+z*z)^8;
                jt=sym(t1)*jss*sym(t4)/(sym(t2)*sym(t3)*sym(t5));
                aas=int(int(jt,x,-x1,x0-x1),y,-y1,y0-y1);
                wq(i)=eval(num2str(eval(char(aas))));
            end
            for i=1:n1,
                ww(j)=sig1(j)+ww(j)+wq(i);
            end
        end
        SSS=0.0;
        for j=1:20
            tq3(j+20*(k-1))=ww(j);
           % 4.3.计算各层土的变形
            ww(j)=ww(j)+sig1(j);%总应力
            ee1(j)=w2(7,k)+sig1(j)/1000/(w3(k,1)+w3(k,2)*sig1(j)/1000);
            ee2(j)=w2(7,k)+ww(j)/1000/(w3(k,1)+w3(k,2)*ww(j)/1000);
            SSS=SSS+(ee1(j)-ee2(j))*H(k)/100/(1+ee1(j));
        end
        S(k)=SSS*100;
    end
      % 5.计算总变形--沉降
    TS=0.0;
    for k=1:n2,
        TS=TS+S(k);
    end
    wer=findobj(gcbf,'tag','edit98');
    set(wer,'string',TS);
```

aaa42.m 代码如下：

```
    % 绘制自重应力与总应力分布图
      tq2=tq2/1000;
      tq3=tq3/1000;
      tt=axes('position',[0.5 0.03 0.35 0.35]);
      plot(tq2,tq1,'g');
      hold on;
      plot(tq3,tq1,'r');
      set(tt,'XAxisLocation','top','XGrid','on','YGrid','on','YDir','reverse');
      tq2=tq2*1000;
      tq3=tq3*1000;
```

激活（单击▶按钮）以后，沉降计算程序运行界面如图 8-19 所示。

图 8-19　沉降计算程序运行界面

习题 8

1．试用 MATLAB 图形窗口编辑 $u\in[-3,2]$， $v\in[-2,3]$ 上 $u^2+2v^2-2=0$ 的图形，将图形标题改为“绘制函数图形”、不显示系统默认菜单、并适当修改其图形与坐标轴标题。

2．简述 MATLAB 中图形用户界面编程时的图形对象类型。

3．制作求解一元二次方程的可视化应用程序（1 个*.fig 文件和 1 个*.m 文件）。

4．简述 MATLAB 中图形用户界面进行可视化编程的一般步骤。

5．试用 MATLAB 图形用户界面编制 MATLAB 环境下的四则运算可视化程序。

6．某工程项目拟使用 MATLAB 平台，开发工程造价分析的可视化应用程序，试阐述编程步骤、开发时可能碰到的问题及其解决方法。

第 9 章　工具箱使用

本章要点：

☑ MATLAB 中的常用工具箱；
☑ MATLAB 中不同优化问题的求解方法；
☑ MATLAB 中神经网络工具箱（主要是 BP 算法）的使用方法；
☑ MATLAB 中小波分析工具箱的使用方法。

本章在简略介绍 MATLAB 工具箱的基础上，简要介绍优化计算工具箱、神经网络工具箱与 BP 算法、小波分析工具箱的理论基础与常用算法。

9.1　MATLAB 工具箱简介

MATLAB 工具箱由一些 m 文件组成，位于安装目录的 Toolboxes 子目录下，用来解决不同领域的专门问题，它集中了全世界研究者在控制系统、信号分析、系统辨识等领域的努力成果。在 MATLAB 说明文件中，列出了这些工具箱的理论基础、示例说明、命令使用方法等内容。在 MATLAB 菜单中选择【Help】→【Full Product Family Help】，在弹出的对话框中选择【Demos】即可显示全部工具箱及有关说明，如图 9-1 所示。表 9-1 以英文首个字母为序、以表格形式列出了常用工具箱的中英文含义与使用说明。

下面简述优化工具箱、神经网络工具箱的一些应用，读者可以结合自己的专业与 MATLAB 的自述帮助文件，学习其他工具箱的使用。

表 9-1　MATLAB 常用工具箱一览表

工具箱英文名称	工具箱中文名称	工具箱使用说明
Communications	通信系统工具箱	创建并分析通信系统
Control System	控制系统工具箱	创建并分析反馈式控制系统
Curve Fitting	曲线拟合工具箱	进行曲线模型的数据拟合与分析
Data Acquisition	数据采集工具箱	由数据采集卡进行数据采集与处理
Financial	金融工具箱	金融数据模型分析，研究金融分析算法
Fuzzy Logic	模糊逻辑工具箱	模糊逻辑系统的设计与仿真分析
Image Processing	图像工具箱	进行图像处理、分析及其算法研究
Instrument Control	仪表工具箱	测试仪表的控制与通信
LMI Control	LMI 控制工具箱	基于优化技术的鲁棒控制器设计
Link for Composer Studio	代码创作室工具箱	基于 MATLAB 与 RTDX 的 Texas 数字信号处理器
Mapping	制图工具箱	信息分析及图形的可视化

续表

工具箱英文名称	工具箱中文名称	工具箱使用说明
Model Predictive Control	模型预测工具箱	约束条件下的大型、多变量问题的控制分析
Mu-Analysis and Synthesis	Mu 分解与合成工具箱	不确定模型体系中多变量反馈式控制器的创建
Neural Network	神经网络工具箱	神经网络的设计与模拟
Optimization	优化工具箱	解决一般大规模优化问题
Partial Differential Equation	偏微分方程工具箱	研究与分析偏微分方程
Robust Control	鲁棒工具箱	多变量反馈式鲁棒控制系统的设计
Signal Processing	信号处理工具箱	信号处理、分析及其算法实现研究
Spline	样条工具箱	数据样条逼近的创建与处理
Statistics	统计工具箱	概率模型与数据统计分析
Symbolic Math	符号数学工具箱	使用符号数学与变量精度控制算法进行计算
System Identification	系统辨识工具箱	根据测得数据创建线性动力模型
Virtual Reality	虚拟现实工具箱	基于 MATLAB 与仿真技术的虚拟现实创建与处理
Wavelet	小波工具箱	使用小波分析技术进行信号分析、压缩和去噪处理

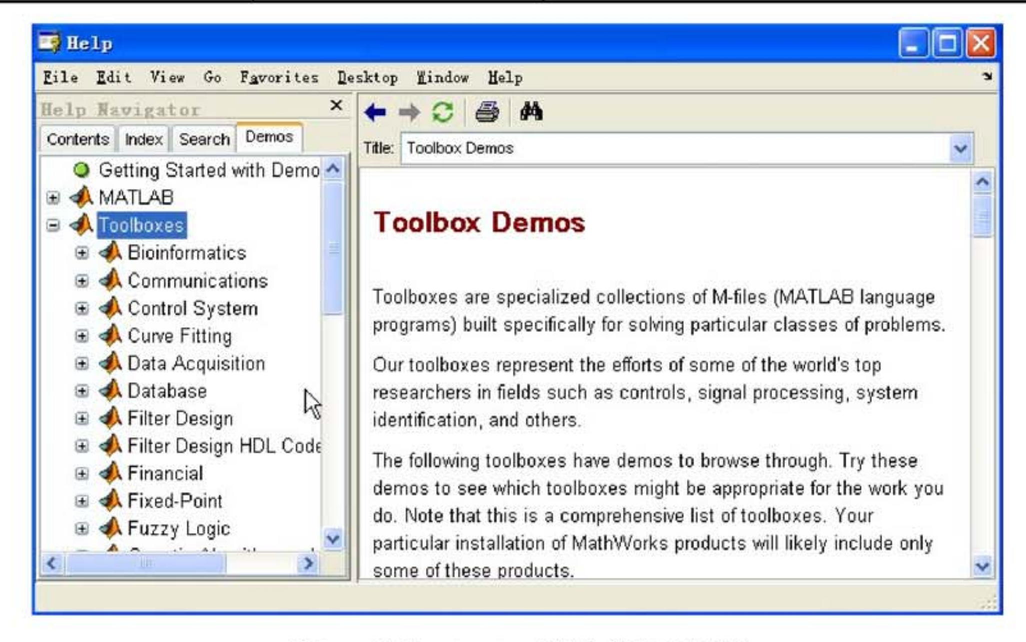


图 9-1 使用 MATLAB 说明文件学习工具箱

9.2 优化工具箱

9.2.1 理论基础

优化工具箱（Optimization Toolbox）位于安装目录下 Toolbox 子目录的 optim 目录之下，用于解决函数的极值问题或设计参数的优化问题。在简单情况下，优化问题可以是随自变量 x 变化系统参数的极大值或极小值；在复杂情况下，是构筑满足不等式约束或参数约束的极值目标函数 $f(x)$。

优化工具箱主要解决如下问题：

（1）无限定条件的非线性极值问题；

（2）限定条件的非线性极值问题，包括目标条件问题、最大最小极值问题、半无穷条件问题；

（3）二次规划和线性规划问题；

（4）非线性最小平方和曲线拟合问题；

（5）非线性方程组求解问题；

（6）限定条件下的线性最小平方问题；

（7）稀疏矩阵和大规模矩阵的极值问题。

在实空间 R^n 内的区域 $x_l \leqslant x \leqslant x_u$，求一组参数 $x=\{x_1,x_2,\cdots,x_n\}$ 使得函数 $f(x)$ 达到极值并且满足等式限制条件 $g_i(x)=0(i=1,\cdots,m_e)$ 和不等式限制条件 $g_i(x)\leqslant 0(i=m_e+1,\cdots,m)$，用数学语言描述，就是：

$$\underset{x\in R^n}{\text{minimize}}\ f(x), s.t. g_i(x)=0(i=1,\cdots,m_e), g_i(x)\leq 0(i=m_e+1,\cdots,m)$$

计算极值可使用搜索法，导数存在时使用梯度法更为有效。

对于二次型问题，$\underset{\mathrm{x}\in R^n}{\text{minimize}}\{f(x)=\frac{1}{2}x^{\mathrm{T}}\boldsymbol{H}x+b^{\mathrm{T}}x+c\}$（$b$ 为常数向量，c 为常数，$\boldsymbol{H}$ 为正定对称矩阵，即 Hessian 矩阵），极值满足 $\nabla f(x^*)=\boldsymbol{H}x^*+b=0$，解为 $x^*=-\boldsymbol{H}^{-1}b$，牛顿法直接计算 $\boldsymbol{H}$ 并使用线性搜索法沿下降方向迭代确定极值，拟牛顿（Quasi-Newton）法通过观测 $f(x)$ 与 $\nabla f(x)$ 的行为建立曲率信息，经过适当更新建立 $\boldsymbol{H}$ 的近似值。计算矩阵 $\boldsymbol{H}$ 的改进方法有 BFGS 方法（H_0 可以是任意对称正定矩阵），其迭代公式为：

$$\boldsymbol{H}_{k+1}=\boldsymbol{H}_k+\frac{q_k q_k^{\mathrm{T}}}{q_k^t s_k}-\frac{\boldsymbol{H}_k^{\mathrm{T}}\boldsymbol{H}_k}{s_k^{\mathrm{T}}\boldsymbol{H}_k s_k},\quad s_k=x_{k+1}-x_k,\quad q_k=\nabla f(x_{k+1})-\nabla f(x_k)$$

求单变量函数极值问题可以使用二次内插的方法；当被计算函数大于 3 或可利用梯度信息时，也可以使用三次内插的方法。

最小二乘问题是使如下的函数平方和达到极小：

$$\underset{x\in R^n}{\text{minimize}}\ F(x)=\underset{x\in R^n}{\text{minimize}}\sum_{i=1}^{m}f_i^2(x)=\underset{x\in R^n}{\text{minimize}}\ f^{\mathrm{T}}(x)f(x)$$

可使用高斯－牛顿(Gauss-New)法或 Levenberg-Marquardt 法求解。高斯－牛顿法中，每一次迭代周期 k 都会得到搜索方向 d_k 并得到一个近似解，而搜索方向使每一次迭代的 $F(x)$ 减少。Levenberg-Marquardt 法搜索方向是线性等式 $(J^{\mathrm{T}}(x_k)J(x_k)+\lambda_k I)d_k=-J(x_k)f(x_k)$ 的解，标量 λ_k 用于控制 d_k 的方向和大小，对于充分大的 λ_k，总使得 $F(x_k+d_k)<F(x_k)$，即每一次迭代中 $F(x)$ 总是逐渐减少的。

对于限定条件下的优化问题，可使用限定条件的罚函数使得限定问题转化为无限定问题求解。

9.2.2　常用命令

（1）FMINCON：用于解决多变量的条件极值问题。其解决的问题具有形式：$\underset{X}{\min}F(X)$。

[常用格式]

[X,FVAL]=FMINCON(FUN,X0,A,B,Aeq,Beq,LB,UB,NONLCON,OPTIONS)

该命令返回目标函数 FUN 在 X 处的极值 FVAL。

X0 是初始值，可以是一个向量或矩阵。

A，B，Aeq，Beq 为线性限定条件 A*X≤B，Aeq**X*=Beq 中的系数矩阵，如果不等式不设定，则 A=[]，B=[]。

LB 和 UB 分别为求解区间 LB≤*X*≤UB 的下限和上限，如果边界不设定、则 LB=[]和/或 UB=[]，如果下限无穷大，则 LB(i)=-Inf，如果上限无穷大，则 UB(i)=Inf。

NONLCON 为 *X* 处的非线性不等式 C(X)≤0 和等式 Ceq(X)=0，该项可以为空。

OPTIONS 是 OPTIMSET 定义的计算选项；如 OPTIONS=[]，则采用系统默认选项。

（2）FMINUNC：用于解决多变量的无条件极值问题。问题形式为：$\min\limits_{X} F(X)$。

[常用格式]

[X,FVAL]=FMINUNC(FUN,X0,OPTIONS)

该命令返回目标函数 FUN 在 *X* 处的无条件极值 FVAL，参数 FUN，*X*0 与 OPTIONS 的意义与命令 FMINCON 中的参数相同，不再赘述。

9.2.3 典型算例

下面使用 MATLAB 7.0 自述文件中所附例子说明优化工具箱的使用方法（为简便起见，引用时作了一些增删），读者也可以通过下面的方式学习：选择【MATLAB 7.0】→菜单窗口【Help】→【Full Product Family Help】→【Demos】→【Toolboxes】→【Optimization】→【Tutorial】，如图 9-2 所示。

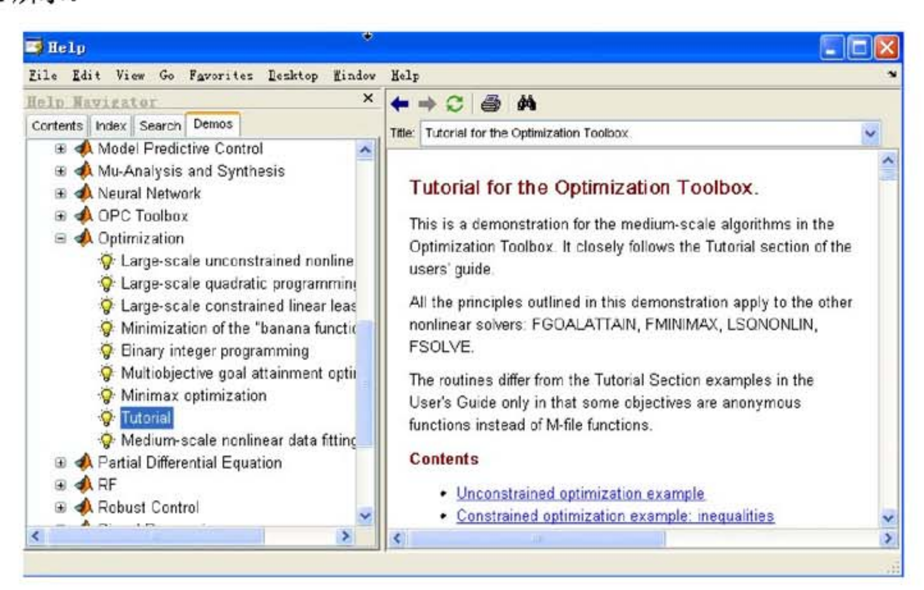


图 9-2 使用 MATLAB 自述说明文件学习优化工具箱

[例 9-1] 无条件极值问题：求函数 $f(X)$ 的极值 $f(X)=\mathrm{e}^{x_1}(4x_1^2+2x_2^2+4x_1x_2+2x_2+1)$

[算例代码]

```
%例 9-1
fun=inline('exp(x(1))*(4*x(1)^2+2*x(2)^2+4*x(1)*x(2)+2*x(2)+1)');
                                                    %定义目标函数
x0=[-1 1];                                          %预设自变量 x=[x1,x2]初始值 x0
options=optimset('LargeScale','off');               %设置为非大规模问题
[x,fval,exitflag,output]=fminunc(fun,x0,options);   %计算极值
x                                                   %显示极值点
fval                                                %显示极值点处的函数值
output.funcCount                                    %显示获得极值点的迭代次数
```

[运行结果]

x =　　0.5000　　-1.0000

fval =　1.0983e-015

ans =　　66

[例9-2]　条件极值问题：求函数 $f(X)=\mathrm{e}^{x_1}(4x_1^2+2x_2^2+4x_1x_2+2x_2+1)$ 在限定条件下的极值，限定条件为 $1.5+x_1x_2-x_1-x_2\le 0,-x_1x_2-10\le 0$。

[算例代码]

```
%例 9-2
fun=inline('exp(x(1))*(4*x(1)^2+2*x(2)^2+4*x(1)*x(2)+2*x(2)+1)');
                                                %定义目标函数
x0 = [-1 1];                                    %预设自变量 x=[x1 x2]初始值 x0
options=optimset('LargeScale','off','Display','off'); %定义为非大规模问题，不显示迭代过程
[x,fval,exitflag,output] = fmincon('objfun',x0,[],[],[],[],[],[],'f0902',options);
x                                               %显示极值点
fval                                            %显示极值点处的函数值
output.funcCount                                %显示获得极值点的迭代次数

%定义 f0902 函数（下述代码另存为工作目录下的 f0902.m 文件）
function [c,ceq]=f0902(x)              %非线性不等式与等式条件极值问题的条件函数
c=[1.5+x(1)*x(2)-x(1)-x(2);-x(1)*x(2)-10];      %非线性不等式条件(矩阵)
ceq=[];                                         %非线性等式条件，为空矩阵
```

[运行结果]

x =　　-9.5474　　1.0474

fval =　　0.0236

ans =　　36

[例 9-3]　边界值条件极值问题：求函数 $f(X)=\mathrm{e}^{x_1}(4x_1^2+2x_2^2+4x_1x_2+2x_2+1)$ 在限定条件下的极值，条件为 $1.5+x_1x_2-x_1-x_2\le 0,\ -x_1x_2-10\le 0;\ x(1)\ge 0, x(2)\ge 0$

[算例代码]

```
%例 9-3
fun=inline('exp(x(1))*(4*x(1)^2+2*x(2)^2+4*x(1)*x(2)+2*x(2)+1)');
```

```
                                            %定义目标函数
x0 = [-1 1];                                %预设自变量 x=[x1 x2]初始值 x0
options=optimset('LargeScale','off','Display','off');  %定义为非大规模问题，不显示迭代过程
lb=zeros(1,2);                              %下边界 X≥0
ub=[];                                      %没有上边界
[x,fval,exitflag,output] = fmincon('objfun',x0,[],[],[],[],[],[],'f0903',options);
x                                           %显示极值点，限定函数仍用前述 confun1
fval                                        %显示极值点处的函数值
[c, ceq] = f0903(x)                         %显示极值点处的约束函数值

%定义 f0903 函数（下述代码另存为工作目录下的 f0903.m 文件）
function [c,ceq]=f0903(x)                   %非线性不等式与等式条件极值问题的条件函数
c=[1.5+x(1)*x(2)-x(1)-x(2);-x(1)*x(2)-10];  %非线性不等式条件(矩阵)
ceq=[];                                     %非线性等式条件，为空矩阵
```

[运行结果]

```
x =     -9.5474     1.0474
fval =      0.0236
c =   1.0e-007 *[     -0.9032      0.9032]
ceq =       []
```

[例 9-4] 使用梯度法求条件极值问题：求函数 $f(X)=e^{x_1}(4x_1^2+2x_2^2+4x_1x_2+2x_2+1)$ 在限定条件下的极值，条件为$1.5+x_1x_2-x_1-x_2\le 0$，$-x_1x_2-10\le 0$

[算例代码]

```
%例 9-4
x0=[-1 1];                                  %预设自变量 x=[x1 x2]初始值 x0
options=optimset('LargeScale','off');
options=optimset(options,'GradObj','on','GradConstr','on');
[x,fval,exitflag,output]=fmincon(' f09041',x0,[],[],[],[],[],[],'f09042',options);
x                                           %显示极值点
fval                                        %显示极值点处的函数值
output.funcCount                            %显示获得极值点的迭代次数

%定义 f09041 函数（下述代码另存为工作目录下的 f09041.m 文件）
function [f,G]= f09041(x)                               %目标函数及其梯度
f=exp(x(1))*(4*x(1)^2+2*x(2)^2+4*x(1)*x(2)+2*x(2)+1);  %目标函数
t=exp(x(1))*(4*x(1)^2+2*x(2)^2+4*x(1)*x(2)+2*x(2)+1);
G=[t+exp(x(1))*(8*x(1)+4*x(2)),exp(x(1))*(4*x(1)+4*x(2)+2)];
%目标函数的梯度(偏导数)

%定义 f09042 函数（下述代码另存为工作目录下的 f09042.m 文件）
function [c, ceq, dc, dceq] = f09042 (x)            %限定条件
c = [1.5 + x(1)*x(2) - x(1) - x(2); -x(1)*x(2) - 10];  %非线性不等式条件
dc = [x(2)-1, -x(2); x(1)-1, -x(1)];                %非线性不等式条件的梯度(偏导数)
ceq = [];dceq = [];                                 %非线性与梯度等式条件,均为空矩阵
```

[运行结果]

x =　-9.5474　1.0474

fval =　0.0236

ans =　18

[例9-5]　等式条件极值问题：求函数 $f(X)=e^{x_1}(4x_1^2+2x_2^2+4x_1x_2+2x_2+1)$ 在限定条件下的极值，条件为 $x_1^2+x_2=1$，$-x_1x_2\le 10$。

[算例代码]

```
%例 9-5
f=exp(x(1)) * (4*x(1)^2 + 2*x(2)^2 + 4*x(1)*x(2) + 2*x(2) + 1);      %目标函数
x0 = [-1 1];                                              %预设自变量 x=[x1 x2]初始值 x0
options =optimset('LargeScale','off');                    %设置为非大规模问题
[x,fval,exitflag,output] = fmincon('objfun',x0,[],[],[],[],[],[],' f0905',options);
x                                                         %显示极值点
fval                                                      %显示极值点处的函数值
[c,ceq] = f0905 (x)                                       %显示极值点处的约束函数值

%定义 f0905 函数（下述代码另存为工作目录下的 f0905.m 文件）
function [c,ceq]= f0905 (x)                               %限定条件
c=-x(1)*x(2)-10;                                          %非线性不等式条件
ceq=x(1)^2+x(2)-1;                                        %非线性等式条件
```

[运行结果]

x =　-0.7529　0.4332

fval =　1.5093

c =　-9.6739

ceq =　6.3038e-009

[例 9-6]　改变默认误差的非限定条件极值问题：求函数 $f(X)$在限定条件下的极值，使得自变量 X误差与目标函数 $f(X)$误差均小于 10^{-8}。 $f(X)=e^{x_1}(4x_1^2+2x_2^2+4x_1x_2+2x_2+1)$。

[算例代码]

```
%例 9-6
fun = inline('exp(x(1)) * (4*x(1)^2 + 2*x(2)^2 + 4*x(1)*x(2) + 2*x(2) + 1)');
x0 = [-1 1];
options =optimset('LargeScale','off');                    %设置为非大规模问题
options = optimset(options,'TolX',1e-8);                  %自变量 X 误差设置
options = optimset(options,'TolFun',1e-8);                %目标函数 F 误差设置
[x,fval,exitflag,output]=fminunc(fun,x0,options);         %计算极值
x                                                         %显示极值点
fval                                                      %显示极值点处的函数值
output.funcCount                                          %显示获得极值点的迭代次数
```

[运行结果]

x =　0.5000　-1.0000

fval =　1.0983e-015

ans = 105

优化工具箱中的非线性曲线拟合命令 lsqnonlin 可用于对指定的非线性函数进行曲线拟合，见 MATLAB 基本应用部分的曲线拟合内容。

9.3 神经网络工具箱

神经网络工具箱（Neural NetWork Toolbox）位于安装目录下 Toolbox 子目录的 nnet 目录中，包括感知器、多层映射 BP 网络、径向基函数（RBF）网络等内容，具有解决数值逼近、模型分析、信号处理、聚类分析、模式分类等功能，可用于工程技术、金融分析、医学研究、人工智能、自动控制、统计分析等领域。下面以不同领域中常用的 BP 算法为例说明有关理论基础及 MATLAB 7.0 中神经网络工具箱的用法。

9.3.1 BP 算法基础

动量 BP 算法，即 Momentum Backpropagation，以其理论严密、适用性强、并且是一种无模型的计算机学习方法，在许多领域得到了应用。如图 9-3 所示，BP 网络常分为三层，即输入层、隐层与输出层，设输入矢量为 $\boldsymbol{X}=(x_1, x_2,\cdots,x_n)^{\mathrm{T}}\in\mathbf{R}^n$，隐层为 $\boldsymbol{X}'=(x'_1,x'_2,\cdots,x'_k)^{\mathrm{T}}\in\mathbf{R}^k$，计算输出为 $Y=(y_1,y_2,\cdots,y_m)^{\mathrm{T}}\in\mathbf{R}^k$，实际输出（也称目标输出）为 $T=(t_1,t_2,\cdots,t_m)^{\mathrm{T}}\in\mathbf{R}^m$，输入层与隐层之间的连接权为 w_{ij}、阈值为 θ_j，隐层与输出层之间的连接权为 w_{jl}、阈值为 θ_l，$i=1$，…，n；$j=1$，…，k；$l=1$，…，m。

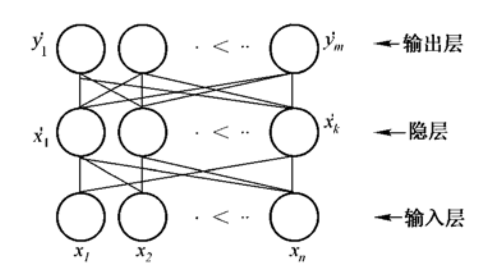


图 9-3 BP 网络

n 维输入向量到 m 维输出向量之间的映照为：

$$y_l = f\left(\sum_{j=1}^{k} w_{jl}x'_i - \theta_l\right),\quad x'_j = f\left(\sum_{i=1}^{n} w_{ij}x_i - \theta_j\right)$$

式中，$f(x)$ 为传递函数，其通常取为 S 形函数，如对数 S 形函数 $f(x)=\dfrac{1}{1+\mathrm{e}^{-x}}$ 和双曲正切 S 形函数 $f(x)=\dfrac{\mathrm{e}^{x}-\mathrm{e}^{-x}}{\mathrm{e}^{x}+\mathrm{e}^{-x}}$。

对于 P 个样本来说，每一样本的误差设定为：

$$\varepsilon = \frac{1}{2}\sum_{l=1}^{m}\left(t_l^{P_1} - y_l^{P_1}\right)^2$$

若采用梯度算法，各层权值的改变量为：

$$\Delta w_{sq} = -\sum_{P_1=1}^{P} \eta \frac{\partial \varepsilon}{\partial w_{sq}}$$

式中，η 为步长，$sq=ij$、jl，P 个样本的总误差为 $E_{总} = \frac{1}{2}\sum_{P_1=1}^{P}\sum_{l}^{m-1}(t_l^{P_1} - y_l^{P_1})^2$。由于总误差的改变量：

$$\Delta E_{总} = \sum_{P_1=1}^{P}\sum_{sq}\frac{\partial \varepsilon}{\partial w_{sq}}\Delta w_{sq} = -\eta\sum_{P_1=1}^{P}\sum_{sq}\left(\frac{\partial \varepsilon}{\partial w_{sq}}\right)^2 \leq 0$$

故这种学习方式总是使总误差向着减少的方向变化，达到一个局部极小解。

对于第 n_0 次替代，输出层—隐层之间连接权的改变量为：

$$\begin{aligned} w_{jl}(n_0+1) - w_{jl}(n_0) &= -\eta\frac{\partial E_{总}}{\partial w_{jl}} = -\eta\sum_{P_1=1}^{P}\frac{\partial E_{P_1}}{\partial y_l^{P_1}}\frac{\partial y_l^{P_1}}{\partial u_l^{P_1}}\frac{\partial u_l^{P_1}}{\partial w_{jl}} \\ &= -\eta\sum_{P_1=1}^{P}(t_l^{P_1} - y_l^{P_1})\times\ y_l^{P_1}(1-y_l^{P_1})\times(-x_j'^{P_1}) \\ &= +\eta\sum_{P_1=1}^{P}\delta_{jl}^{P_1}x_j'^{P_1} \end{aligned}$$

式中，$u_l^{P_1} = -\sum_{j=1}^{k} w_{jl}x_j'^{P_1}$，$\delta_{jl}^{P_1} = (t_l^{P_1} - y_l^{P_1})\times\ y_l^{P_1}(1-y_l^{P_1})$。

而第 n_0 次替代时，隐层—输入层之间连接权的改变量为：

$$\begin{aligned} & w_{ij}(n_0+1) - w_{ij}(n_0) = -\eta\frac{\partial E_{总}}{\partial w_{ij}} \\ &= -\eta\sum_{P_1=1}^{P}\sum_{l=1}^{m}\frac{\partial E_{P_1}}{\partial y_l^{P_1}}\frac{\partial y_l^{P_1}}{\partial u_l^{P_1}}\frac{\partial u_l^{P_1}}{\partial x_j'^{P_1}}\frac{\partial x_j'^{P_1}}{\partial v_j^{P_1}}\frac{\partial v_j^{P_1}}{\partial w_{ij}} \\ &= -\eta\sum_{P_1=1}^{P}\sum_{l=1}^{m}(t_l^{P_1} - y_l^{P_1})\times\ y_l^{P_1}(1-y_l^{P_1})\times\ w_{jl}\times[1-(x_j'^{P_1})^2]\times(-x_i^{P_1}) \\ &= +\eta\sum_{P_1=1}^{P}\sum_{l=1}^{m}(t_l^{P_1} - y_l^{P_1})\times\ y_l^{P_1}(1-y_l^{P_1})\times\ w_{jl}[1-(x_j'^{P_1})^2]x_i^{P_1} \\ &= \eta\sum_{P_1=1}^{P}\sum_{l=1}^{m}\delta_{jl}^{P_1}\times\ w_{jl}[1-(x_j'^{P_1})^2]\times\ x_i^{P_1} \\ &= \eta\sum_{P_1=1}^{P}\delta_{ij}^{P_1}x_i^{P_1} \end{aligned}$$

式中，$v_j^{P_1} = -\sum_{i=1}^{n} w_{ij}x_i^{P_1}$，$\delta_{ij}^{P_1} = \sum_{l=1}^{m}\delta_{jl}^{P_1}\times\ w_{jl}[1-(x_j'^{P_1})^2]$。

为了加速收敛并防止振荡，可以在权值改变量中加入一个动量因子 α，$0<\alpha<1$，即

$$w_{sq}(n_0+1)=w_{sq}(n_0)+\eta\sum_{P_1=1}^{P}\delta_{sq}^{P_1}x_s^{P_1}+\alpha[w_{sq}(n_0+1)-w_{sq}(n_0)],\quad sq=ij,jl$$

对于第n_0步迭代，如果各样本误差与总误差没有达到预定要求，则连接权调整为：

$$w_{jl}(n_0+1)=w_{jl}(n_0)+\eta\sum_{P_1=1}^{P}\delta_{jl}^{P_1}x_j^{P_1}+\alpha\Delta w_{jl}(n_0)$$

$$w_{ij}(n_0+1)=w_{ij}(n_0)+\eta\sum_{P_1=1}^{P}\delta_{ij}^{P_1}x_i^{P_1}+\alpha\Delta w_{ij}(n_0)$$

不断重复上述步骤，即可得到符合训练样本要求的权值与阈值，对于一个待求样本参数，可以使用调整后的权值与阈值分析与预测。

9.3.2 BP 算法常用命令

动量 BP 算法通过使用不同的传递函数训练多层反馈式网络（虽然多层网络最为常用，但也可用其他类型的网络训练），可用于函数逼近、聚类分析、模式识别等方面的研究。反向传播一词，与计算网络误差（用于计算网络权重与偏量）函数导数过程有关。多层网络并不完全受问题所限。虽然输入参数和输出神经元受样本数所限，但网络输入输出之间的层数与大小可由用户确定。如果 S 层具有足够多的单元，两层 S/线性网络可以模拟输入输出之间的任何函数关系。BP 训练方法有多种，每一方法又有不同的计算要求和容量要求，没有一个适用于所有问题的算法。为使数据输入输出更好，MATLAB 提供了一些前后处理函数，见表 9-2。表 9-3 为 MATLAB 工具箱中的训练算法列表。

表 9-2 MATLAB 中 BP 网络的前后处理函数

函　数	简　介
premnmx	使数据归一化于区间[-1,1]
postmnmx	premnmx 的逆过程，使数据返回到原来单位
tramnmx	用最大最小值法使已有数据归一化，可用于 premnmx 训练网络的新数据处理
prestd	数据归一化、使其具有零均值和单位标准差
poststd	prestd 的逆过程，使数据返回为到原来单位
trastd	对已有计算均值和标准差数据进行归一化处理，用于 prestd 命令归一化训练网络的新数据处理
prepca	主因子分析，可减少输入向量大小并消除其中的非相关向量
trapca	用已有主因子转换矩阵对数据进行处理，由 prepca 命令进行数据转换训练的网络对新数据处理
postreg	网络输出和目标值的线性返回值，用于确定网络的适应性

表 9-3 MATLAB 中 BP 网络训练函数

函　数	简　介
traingd	基本梯度下降算法，灵敏度较差，可用于增量模型训练
traingdm	带动量的梯度下降算法，训练速度比 traingd 一般要快，可用于增量模型训练
traingdx	自适应学习准则，训练速度比 traingd 一般要快，但只能用于整批模型训练

续表

函　数	简　介
trainrp	松弛反向传播算法，一种简单批量模型训练算法，收敛较快、所需容量较小
traincgf	Fletcher-Reeves 共轭梯度算法，是一种具有较小容量需求的共轭梯度算法
traincgp	Polak-Ribiére 共轭梯度算法，容量需求比 traincgf 稍大、但对一些问题收敛更快
traincgb	Powell-Beale 共轭梯度算法，容量需求比 traincgp 稍大，收敛很快
trainscg	比例型共轭梯度算法，不需要线性搜索，是很好的一般目标训练算法
trainbfg	BFGS 算法（改进的拟牛顿法），与共轭梯度算法相比，其需要储存相当的 Hessian 矩阵、每一迭代计算量要大一些，但达到收敛的迭代步数通常要少一些
trainoss	单步余弦算法，共轭梯度算法和拟牛顿算法的折中方法
trainlm	Levenberg-Marquardt 算法，对于一般大小网络训练算法快，当训练样本较多时具有内存折减特性
trainbr	Bayesian 规则，是 Levenberg-Marquardt 训练算法的改进方法、使产生的网络更好一些，在确定优化网络时困难会少一些

下面介绍一些常用命令。

（1）nnt2ff

[调用格式]

```
net = nnt2ff(pr,{w1 w2…},{b1 b2…},{tf1 tf2…},btf,blr,pf)
```

nnt2ff 返回一个反馈式网络。

PR 是 R 个输入数的最大值与最小值矩阵 $\boldsymbol{R}\times 2$。

Wi 是第 i 层的权重矩阵。

Bi 是第 i 层的偏量向量。

Tfi 是第 i 层的传递函数，默认值为'tansig'。

BTF 是反向传播网络的训练函数，可为下述函数之一：'traingd'、 'traingdm'、 'traingda'、'traingdx'、'trainlm' (默认值为'traingdx')。

BLF 为权重与偏量反向传播的学习函数，可为下述函数之一：learngd、learngdm(默认值为'learngdm')。

PF 为显示函数，可以是下述函数之一：mse、mseerg(默认值为'mse')。

（2）netsum

[调用格式]

```
N=netsum(Z1,Z2,…)
df=netsum('deriv')
```

NETSUM 是网络输入函数集，包括输入权重和偏量；Z1，Z2，…，Zn 为输入数据；N=netsum(Z1,Z2,…,Zn)返回 n 个输入。NETSUM('deriv')返回 NETSUM 的导数。

（3）newff

[调用格式]

```
net = newff(PR,[S1 S2…SNl],{TF1 TF2…TFNl},BTF,BLF,PF)
```

newff 创建一个带有对话框的反馈式网络 net。

PR 为 R 个输入数的最大值与最小值矩阵 $\boldsymbol{R}\times 2$。Si 为 N1 层中第 i 层的大小。

Tfi 是第 i 层的传递函数，可为'tansig'、'logsig'、'purelin'之一，默认值为'tansig'。

BTF 是反向传播网络的训练函数,可为'trainlm'、'trainbfg'、'trainrp'、'traingd'、'learngdm'

之一，默认值为'trainlm'。

BLF 为权重与偏量反向传播的学习函数，可为'learngd'、'learngdm'之一，默认值为'learngdm'。

PF 为显示函数，可以是下述函数之一：'mse'、msereg'，默认值为'mse'。

（4）init

[调用格式]

```
M=init(M0,R,PAR,SP)
```

M0 为初始网络；M 为采用新初始参数值后的网络；R 为随机初始参数变量；PAR 为初始参数平均值；SP 为系统稳定参数，SP='p'则预测稳定，SP='s'则系统稳定，SP='b'则为预测与系统稳定，默认值为 SP='p'。

（5）learngdm

[调用格式]

```
[dW,LS] = learngdm(W,P,Z,N,A,T,E,gW,gA,D,LP,LS)
```

learngdm 为使用含动量的梯度下降法学习函数，返回改变后的权值矩阵 dW(大小为$S\times R$)和新学习状态值 LS。

M 为采用新初始参数值后的网络。

W 为权值矩阵($S\times R$)或偏值向量($S\times 1$)。

P 为输入向量($R\times Q$)。

Z 为权值输入向量($S\times Q$)。

N 为网络输入向量($S\times Q$)。

A 为输出向量($S\times Q$)。

T 为各层目标向量($S\times Q$)。

E 为各层误差向量($S\times Q$)。

gW 为显示梯度值($S\times R$)。

gA 为显示输出梯度($S\times Q$)。

D 为神经元距($S\times S$)。

LP 为学习参数，如使用系统默认值，则 LP=[]。

LS 为学习状态，初始值应为空，即 LS =[]。

（6）logsig

[调用格式]

```
A=logsig(N)
```

logsig 由 N 个网络输入元素计算网络输出的传递函数值 A，数学含义是 logsig(N)=1/(1+exp(-N))。

（7）network

[调用格式]

```
net=network(numInputs,numLayers,biasConnect,inputConnect,
    layerConnect,outputConnect,targetConnect)
```

net 为新产生的神经网络。

M 为采用新初始参数值后的网络。

numInputs 为输入单元数(默认值为 0)。

numLayers 为网络层数(默认值为 0)。

biasConnect 为布尔向量(默认值为 0 向量)，大小为 numLayers×1。

inputConnect 为布尔矩阵(默认值为 0 矩阵)，大小为 numLayers×numInputs。

layerConnect 为布尔矩阵(默认值为 0 矩阵)，大小为 numLayers×numLayers。

outputConnec 为布尔向量(默认值为 0 向量)，大小为 1×numLayers。

targetConnect 为布尔向量(默认值为 0 向量)，大小为 1×numLayers。

（8）purelin

[调用格式]

```
A=purelin(N)
```

A=purelin(N)返回 N 个网络输入元素计算网络输出的传递函数值 A。

（9）sim

[调用格式]

```
[T,X,Y1,…,Yn]=sim('model',TIMESPAN,OPTIONS,UT)
```

sim 是一个模型模拟函数，如为 sim[]则采用系统默认设置。

T 为返回时间向量。

X 为矩阵或结构数组形式返回的状态。

Y 为矩阵或结构数组形式返回的输出结果。

Y1,…,Yn 仅适用于块体模型，n 是块体数目，Y1,…,Yn 是返回变量。

'model'为块体模型名称。

TIMESPAN 为时间间隔。

OPTIONS 为模拟参数设置。

UT 为外部输出选择，可选或不选。

（10）tansig

[调用格式]

```
A=tansig(N)
```

tansig 是由 N 个网络输入元素计算网络输出传递函数（双曲正切型 S 函数）的函数值 A，数学含义是 tansig(N)=2/(1+exp(-2*N))-1。

（11）train

[调用格式]

```
[net1,tr,Y,E,Pf,Af]=train(net,P,T,Pi,Ai,VV,TV)
```

train 是 net.trainFcn 和 net. trainParam 定义的神经网络训练函数。

net，net1 分别为旧神经网络和新神经网络。

tr 为单步训练记录。

Y 为网络输出。

E 为网络训练误差。

Pf 为最后训练情况。

Af 为最末层情况。

P 为网络输入。

T 为网络目标输出。

Pi 为初始输入情况。

Ai 为各层初始输入情况。

VV 为确认向量的结构数组，默认值为空矩阵[]。

TV 为验证向量的结构数组，默认值为空矩阵[]。

（12）trainlm

[调用格式]

[net1,tr]=trainlm(net,Pd,Tl,Ai,Q,TS,VV,TV)

trainlm 是 Levenberg-Marquardt 反向传播优化算法中改变权值和偏量的网络训练函数。

net，net1 分别为旧神经网络和新神经网络。

tr 为单步训练记录。

Pd 为输入向量。

Tl 为输出向量。

Ai 为初始输入情况。

Q 为样本大小。

TS 为时间步。

VV，TV 为空矩阵[]或确认向量的结构数组。

动量 BP 算法的计算步骤如下。

步骤 1：输入训练样本输入参数$\boldsymbol{p}$、目标参数t 与试验样本输入参数$\boldsymbol{p}_1$；

步骤 2：建立网络，根据算法不同、使用命令为 newff 等；

步骤 3：训练，即根据初始权重与偏量、输入与输出样本、隐单元数、传递函数，获得满足训练参数的调整后的权值与偏量，这是整个算法中的关键步骤；

步骤 4：仿真，即根据调整后的权重与偏量重新计算训练样本与试验样本的输出值，

MATLAB 中动量 BP 算法的一般模型和基本原理等内容，参见 MATLAB 7.0 自述文件，选择【MATLAB 7.0】→菜单窗口【Help】→【Full Product Family Help】→【Contents】→【Neural Network Toolbox】→【Backpropagation】，如图 9-4 所示。其他有关神经网络的内容，也可使用相似的方式学习和使用。

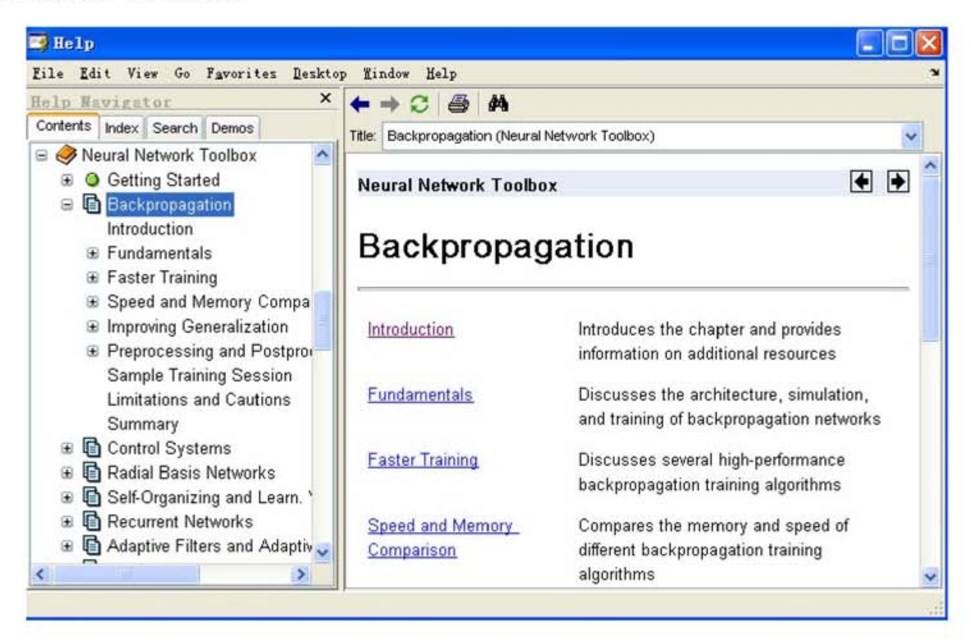


图 9-4 使用 MATLAB 自述说明文件学习神经网络工具箱

9.3.3　BP 算法算例

下面以动量 BP 算法的使用实例说明其 MATLAB 平台下的编程方法。

[例 9-7]　已知 7 个样本输入参数为 p=[1.0 4;1.5 2;1.5 6;2.5 4;2.0 4;3.0 8;3.0 4]，各样本对应的输出参数为 t=[0.1581;0.3261;0.2812;0.4755;0.4107;0.5650;0.6481]，试用 2×8×1 动量 BP 网络(输入层、隐层、输出层单元数分别为 2、8、1)预测新样本 p1=[2.5 4]的输出结果。

[算例代码]

```
%例 9-7
tic                                                          %开始计时
%输入参数
    %p 中 1.0,4 为第 1 个样本输入参数(输入层单元为 2,即分号相隔的数据量,其余类推)
    %t 中 0.1581 表示第 1 个样本输出结果(输出层单元数为 1)
    %p1=[2.5 4]为待确定样本,yc 为实测结果
p0=[1.0 4;1.5 2;1.5 6;2.5 4;2.0 4;3.0 8;3.0 4];              %输入参数，注意输入方法
p=p0';
t=[0.1581 0.3261 0.2812 0.4755 0.4107 0.5650 0.6481];        %目标输出
net=newff(minmax(p),[3,1],{'tansig','purelin'},'trainlm');   %L-M 方法建立网络
[net,tr]=train(net,p,t);                                     %训练网络
p1=[2.5;4];                                                  %待确定样本
a=sim(net,p1)                                                %网络仿真结果
toc                                                          %计时结束
```

[运行结果]

动量 BP 算法模拟情况如图 9-5 所示，由图可知，达到预测结果只需 9 步，预测值 a 和所用时间 elapsed_time(秒)分别为（由于网络初始值不同，所需时间与下述结果可能不同）：

```
a =      0.4755
elapsed_time =      3.234000
```

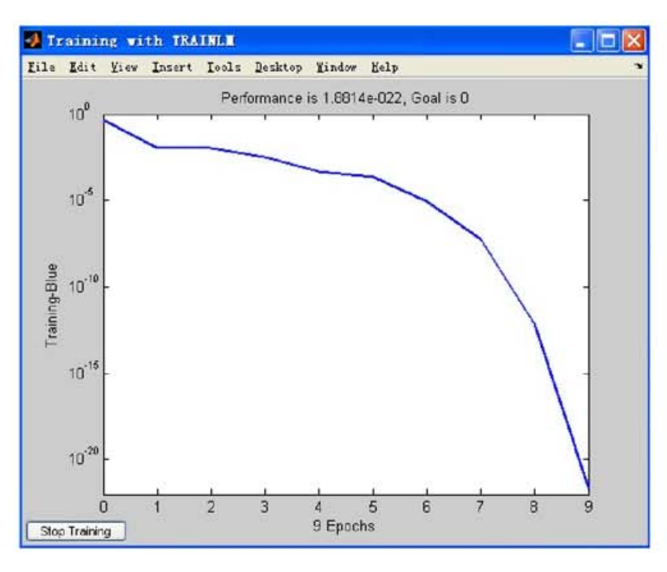


图 9-5　动量 BP 算法模拟情况

9.4 小波分析工具箱

小波(wavelet)，即小区域的波，是一种特殊的、长度有限、平均值为 0 的波形，在时域都具有紧支集或近似紧支集，同时具有正负交替的“波动性”。小波分析(wavelet analysis)和小波变换是傅里叶（Fourier）变换的一个重大突破与发展，与傅里叶变换、窗口傅里叶变换（Gabor 变换）相比，它能更有效地从各种信号中提取重要信息，并通过伸缩和平移等运算功能对函数或信号进行多尺度细化分析(Multiscale Analysis)，从而提供了一种自适应、时域和频域同时局部化的分析方法，无论分析低频或高频局部信号，它都能自动调节时-频窗，以适应实际分析的需要。傅里叶变换运用一系列不同频率的正弦波叠加成信号，而小波分析则用一系列的小波函数叠加信号。小波分析的特点是压缩比高、压缩速度快，压缩后能保持信号与图像特征不变、且在传递中可以抗干扰，同时，小波比正弦波更不规则、更不对称，对信号有突变或要表现信号局部特性时，用小波函数来逼近信号比正弦波要好得多。小波分析方法在许多领域（如信号处理、计算机视觉、图像处理、语音分析与合成、地震勘测、土木结构损伤检测、通信与电子系统等）具有广泛的应用。

9.4.1 概述

小波工具箱(Wavelet Toolbox)位于安装目录 Toolbox 中的 wavelet 子目录之下，MATLAB 中常用的小波分析函数见表 9-4，用户自己也可添加新的小波函数来丰富小波函数库。

表 9-4 MATLAB 7.0 中小波分析常用函数

函数命令		函数意义
通用函数	biorfilt	双正交小波滤波器组
	centfrq	计算小波中心频率
	dyaddown	二元抽样
	dyadup	二元插值
	intwave	积分小波函数ψ
	orthfilt	正交小波滤波器组
	qmf	镜像二次滤波器
	scal2frq	计算频率尺度
	wavefun	小波函数和尺度函数
	wavefun2	二维小波函数和尺度函数
	wavemngr	小波管理
	wfilters	小波滤波器
	wmaxlev	计算小波分析的最大尺度
小波函数	biorwavf	双正交样条小波变换
	cgauwavf	复 Gauss 小波
	cmorwavf	复 Morlet 小波
	coifwavf	Coiflets 小波滤波器
	dbaux	Daubechies 小波辅助函数

续表

函 数 命 令		函 数 意 义
小波函数	dbwavf	Daubechies 小波滤波器
	fbspwavf	计算复频 B 样条小波
	gauswavf	Gauss 小波
	mexihat	墨西哥帽小波
小波函数	meyer	Meyer 小波
	meyeraux	Meyer 小波辅助函数
	morlet	Morlet 小波
	rbiowavf	双正交样条小波滤波器的求逆
	shanwavf	复 Shannon 小波
	symaux	Symlets 小波辅助函数
	symwavf	Symlets 小波滤波器
一维小波变换\二维离散小波变换	cwt	一维连续小波变换
	pat2cwav	由预定义的类型创建小波
	appcoef/appcoef2	提取小波变换低频系数/提取二维小波分解低频系数
	detcoef/detcoef2	提取小波变换低频系数/提取二维小波分解高频系数
	dwt/dwt2	单尺度一维离散小波变换/单尺度二维离散小波变换
	dwtmode	离散小波变换拓展格式
	idwt/idwt2	单尺度离散小波逆变换/单尺度二维离散小波逆变换
	upcoef/upcoef2	一维小波系数的直接重构/二维小波系数的直接重构
	upwlev/upwlev2	单尺度一维小波变换的重构/二维小波分解的单尺度重构
	wavedec/wavedec2	多尺度一维小波分解/多尺度二维小波分解
	waverec/waverec2	多尺度一维小波重构/多尺度二维小波重构
	wenergy/wenergy2	计算一维小波分解的能量/计算二维小波分解的能量
	wrcoef/wrcoef2	一维小波系数单支重构/二维小波系数单支重构
小波包算法	bestlevt	计算完整最佳小波包树
	besttree	计算最佳树
	entrupd	更新小波包的熵值
	wentropy	计算小波包的熵
	wp2wtree	从小波包树中抽取小波树
	wpcoef	计算小波包系数
	wpcutree	剪切小波包分解树
	wpdec	一维小波包分解
	wpdec2	二维小波包分解
	wpfun	小波包函数
	wpjoin	重新组合小波包
	wprcoef	小波包分解系数的重构
	wprec	一维小波包分解的重构
	wprec2	二维小波包分解的重构
	wpsplt	分解小波包

MATLAB 7.0 自述文件中以示例说明了小波工具箱的使用方法，可通过下面的方式实现：在MATLAB 菜单中选择【Help】→【Full Product Family Help】→【Demos】→【Toolboxes】→【Wavelet】，得到如图 9-6 所示的示例，在该图中单击一维小波变换【Wavelet 1-D】等位置，可以学习小波分析有关示例。

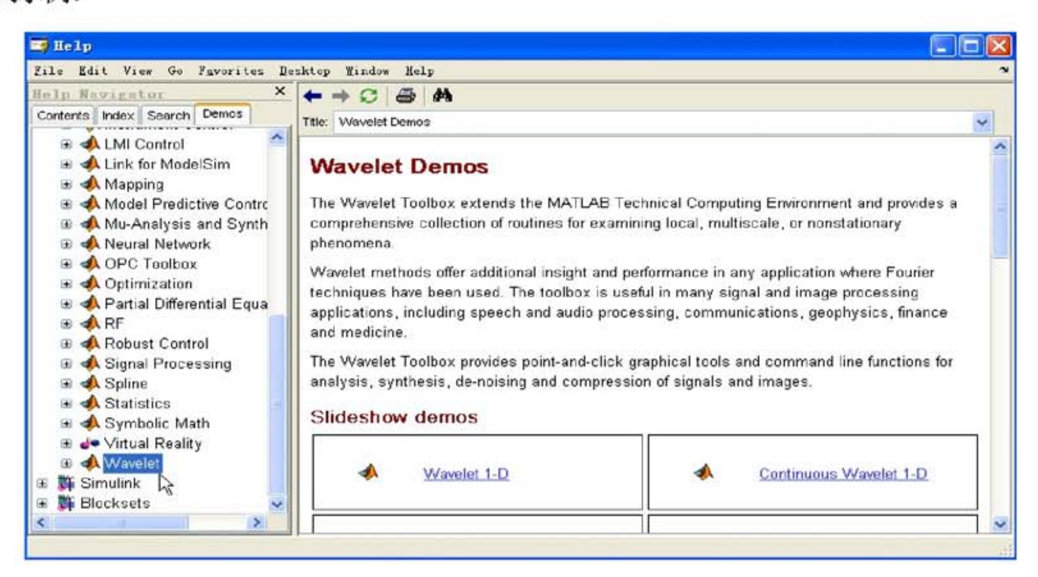


图 9-6 MATLAB 自述文件中的小波分析示例

9.4.2 小波分析方法

使用 MATLAB 语言进行小波分析，可以有两种基本方式：命令方式和图形窗口方式。下面分别介绍。

1．命令方式

小波分析不仅可用于一维信号的处理，也可应用于二维图形等的分析，下面是一维小波变换中较为常用的命令。

（1）CWT

用于实现一维连续(复)小波的分析。

[常用格式]

```
COEFS = CWT(S,SCALES,'wname','plot')
```

COEFS 为连续小波变换后的返回系数矩阵，S 为待分析的信号，SCALES 为小波变换的尺度向量，wname 为小波函数名，小波函数名为 cgau 或 cmor 时为一维连续复小波分析，plot 用来画出小波变换后的系数矩阵 COEFS。

（2）DWT

用于实现单尺度一维离散小波的分析。

[常用格式]

```
[CA,CD] = DWT(X,'wname')
```

X 为待分析的离散信号， wname 为小波函数名，CA、CD 分别为变换后返回的低频系数和高频系数向量。

（3）UPCOEF

是一维系数的直接小波重构函数。

[常用格式]

Y = UPCOEF(O,X,'wname')

X 为待分析的离散信号， wname 为小波函数名，*O* 为 *a* 或 *d* 中之一，*O*=*a* 则对低频系数进行重构；*O*=*d* 则对高频系数进行重构。

（4）WAVEDEC

用于实现多尺度一维离散小波的分析。

[常用格式]

[C,L] = WAVEDEC(X,N,'wname')

X 为待分析的离散信号，*N* 为尺度、其值为正整数，wname 为小波函数名，*C* 由[cA_j, cD_j, cD_{j-1},⋯, cD_1]组成，*L* 由[cA_j 的长度，cD_{j-1} 的长度，cD_{j-1} 的长度，⋯ cD_1 的长度，*X* 的长度]组成。

（5）APPCOEF

用于提取一维低频系数。

[常用格式]

A = APPCOEF(C,L,'wname',N)

N 为需提取的层，值为正整数，wname 为小波函数名，*C* 由[cA_j,cD_j,cD_{j-1},⋯,cD_1]组成，L 由[cA_j 的长度，cD_j 的长度，cD_{j-1} 的长度，cD_1 的长度，⋯ *X* 的长度]组成。

（6）DETCOEF

用于用于提取一维高频系数。

[常用格式]

D = DETCOEF(C,L,N)

N 为需提取的层（正整数），*C* 由[cA_j,cD_j,cD_{j-1},⋯,cD_1]组成，L 由[cA_j 的长度，cD_j 的长度，cD_{j-1} 的长度，cD_1 的长度，⋯ X 的长度]组成。

（7）WRCOEF

用于实现对一维小波系数进行单层重构。

[常用格式]

X = WRCOEF('type',C,L,'wname',N)

N 为正整数，wname 为小波函数名，*C* 由[cA_j,cD_j,cD_{j-1},⋯,cD_1]组成，*L* 由[cA_j 的长度，cD_j 的长度，cD_{j-1} 的长度，cD_1 的长度，⋯ *X* 的长度]组成，type 为 *a* 或 *d* 中之一，type=*a* 则对信号的低频部分进行重构，type=*d* 则对信号的高频部分进行重构。

2．图形窗口方式

使用图形窗口方式进行小波分析，是在 MATLAB 命令窗口键入命令 wavemenu，出现小波工具箱菜单，如图 9-7。由图 9-7 可知，主菜单包括一维分析(One-Dimensional)、一维专门分析(Specialized 1-D)、二维分析(Two-Dimensional)、二维专门分析(Specialized 2-D)、小波分析演示(Display)、小波设计(Wavelet Design)、小波拓展(Extension)等六个部分，每一部分又包括不同的处理方式。

单击【Wavelet 1-D】（以最简单的一维离散小波分析为例，其他分析方法与此相似），出现一维离散小波分析窗口，如图 9-8 所示。

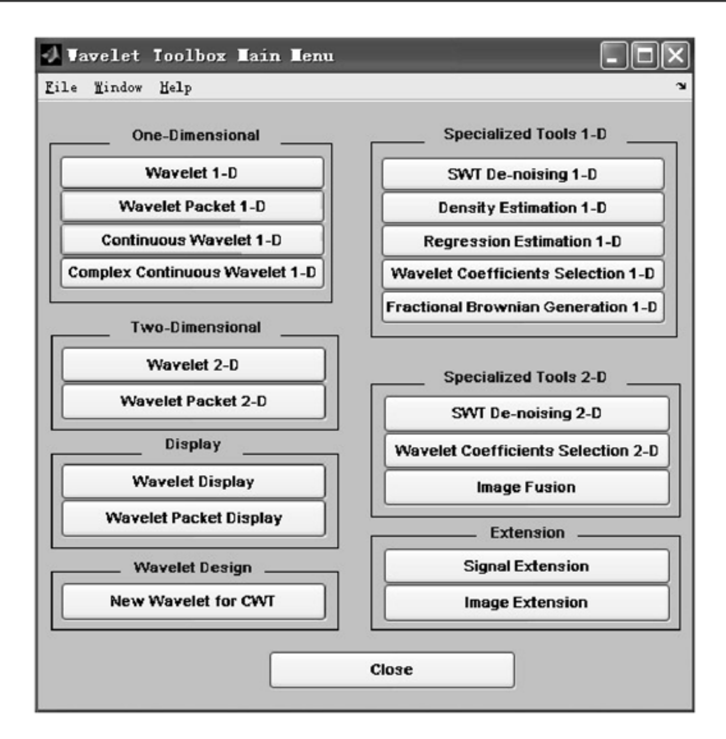


图 9-7 小波工具箱主菜单窗口

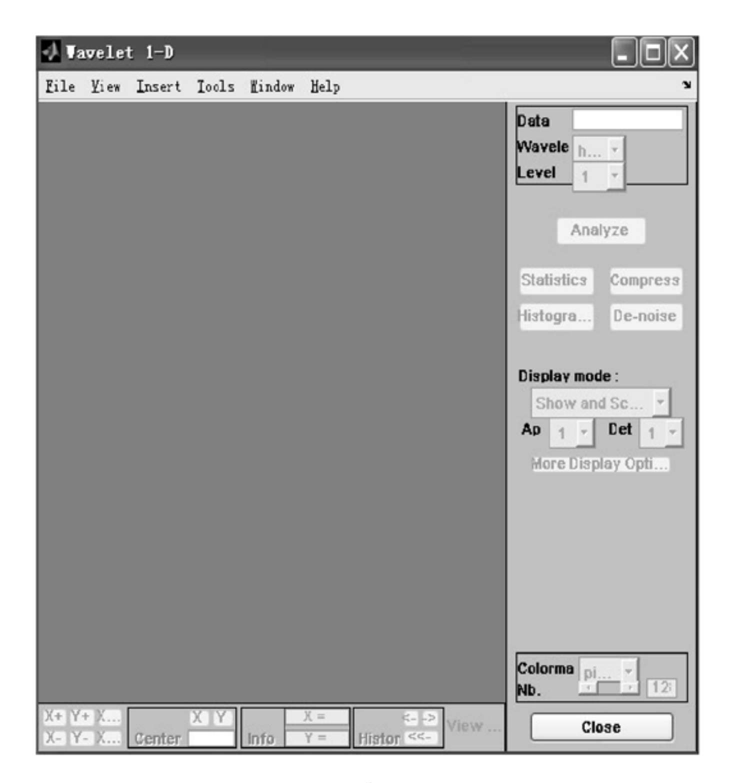


图 9-8 一维离散小波分析窗口

在该窗口中，选择【File】→【Load】→【Signal】，如图 9-9 所示，在弹出的【Load Signal】窗口（如图 9-10 所示）中，选择信号文件(MAT 文件)。比如选择 MATLAB 安装目录下的信

号 toolbox/wavelet/wavedemo/sumsin，然后单击【打开】按钮，此后出现小波分析窗口（如图 9-11 所示）。

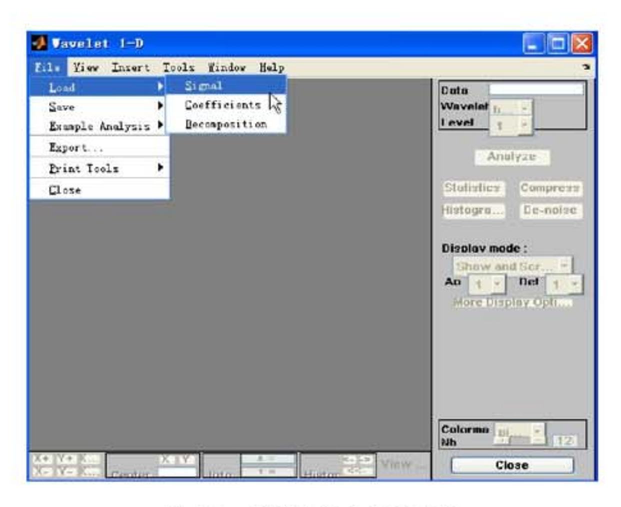


图 9-9　离散小波分析主菜单



图 9-10　选择信号文件对话框

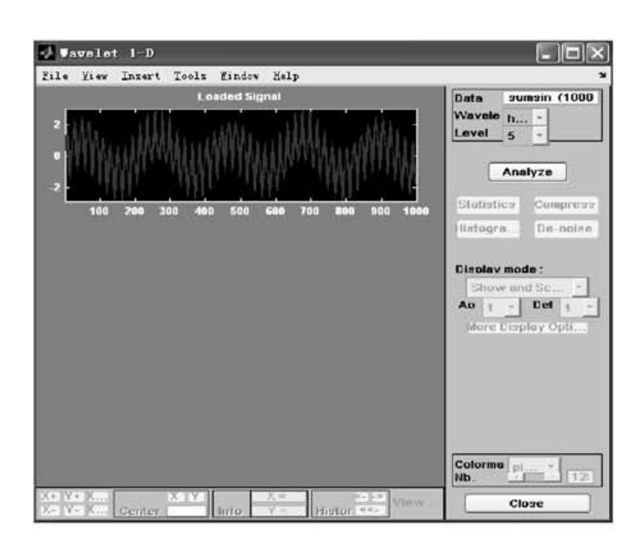


图 9-11　小波分析对话框

在小波分析窗口的下拉列表框【Wavelet】中选择小波函数，在下拉列表框【Level】中选择小波变换尺度，再单击分析按钮【Analyze】，系统得到分析结果，如图 9-12 所示。

对需要重点观察的部位，可在如图 9-12 所示的分析结果窗口的下部用鼠标左键拖框，通过选择 X+（X 轴方向放大）、Y+（Y 轴方向放大）、X-（X 轴方向缩小大）、Y-（Y 轴方向缩小）、XY+（XY 轴同时放大）、XY-（XY 轴同时缩小）进行更细致的分析。

进一步选择【Statistics】、【Compress】等按钮，可得到统计、压缩等方面的特性，如图 9-13 所示。

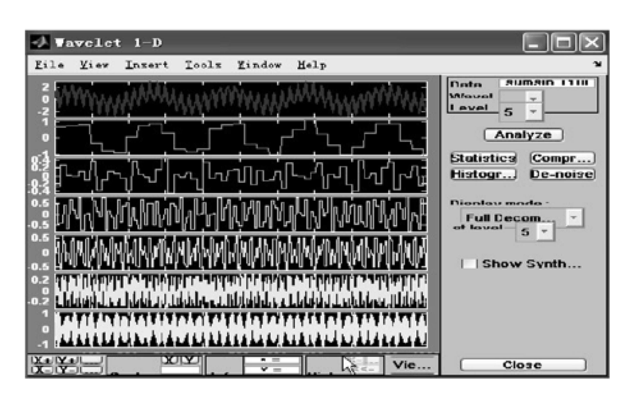


图 9-12　一维离散小波分析结果

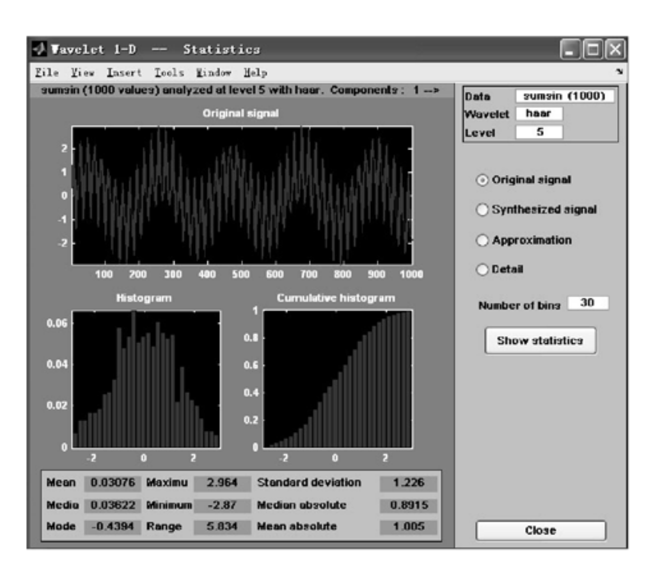


图 9-13　一维离散小波统计分析结果

9.4.3　小波分析典型算例

[例 9-8]　举例说明一维连续小波分析。

[算例代码]

```
%例 9-8
t = linspace(-1,1,100);            %定义向量 t
s = 1-abs(t);                      %信号以 0 为界左右对称
m=[s,s,s,s,s];                     %构造新的信号
plot(m);                           %显示信号
c = cwt(m,1:32,'db4','plot');      %对信号进行小波分析并画出处理后所得的矩阵向量 C
```

[运行结果]

运行结果如图 9-14 所示，它是信号处理后显示的系数图形，色深说明系数值大、色浅说明系数值小。由图 9-14 可知，该尺度下信号周期性比较明显。

[例 9-9]　举例说明一维连续复小波分析。

[算例代码]

```
%例 9-9
t = linspace(-1,1,100);              %定义向量 t
s = 1-abs(t);                        %信号以 0 为界左右对称
m=[s,s,s,s,s];                       %构造新的信号
plot(m);                             %显示信号
c = cwt(m,1:32,'cgau4','plot');      %对信号进行小波分析并画出处理后所得的矩阵向量 C
```

[运行结果]

运行结果如图 9-15 所示，该图是对信号 s 的连续复小波变换，其结果系数分别对应为左上的实部、右上的虚部、左下的模和右下的位相四个部分。

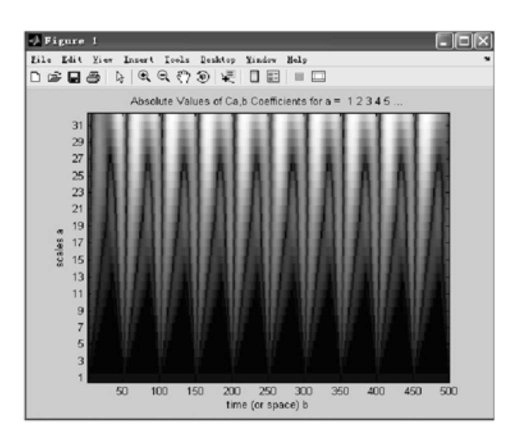


图 9-14　一维连续小波分析简例

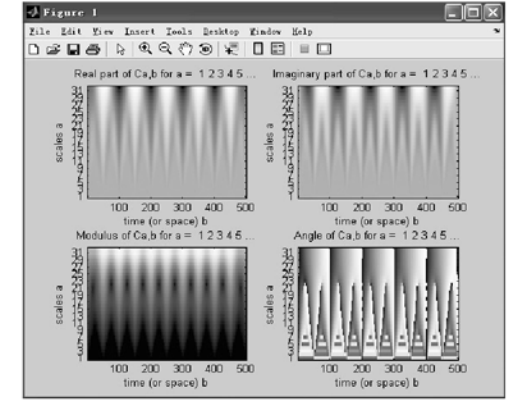


图 9-15　一维连续复小波分析

[例 9-10]　对信号 mishmash 进行一维离散小波分析。

[算例代码]

```
%例 9-10
load mishmash                                     %载入信号 mishmash
S=mishmash(1:1000);                               %构造长度为 1000 的新信号
plot(S);                                          %显示信号曲线
[c,l]=wavedec(S,3,'db2');                         %对信号 S 作 3 尺度的一维分解
a3=wrcoef('a',c,l,'db2',3);                       %重构第三层的低频信号
d1=wrcoef('d',c,l,'db2',1);                       %重构第一层的高频信号
d2=wrcoef('d',c,l,'db2',2);                       %重构第二层的高频信号
d3=wrcoef('d',c,l,'db2',3);                       %重构第三层的高频信号
figure                                            %打开新的图形窗口
subplot(2,2,1);plot(a3);title('近似低频  a3');    %画出信号 a3
subplot(2,2,2);plot(d1);title('细节高频  d1');    %画出信号 d1
subplot(2,2,3);plot(d2);title('细节高频  d2');    %画出信号 d2
subplot(2,2,4);plot(d3);title('细节高频  d3');    %画出信号 d3
```

[运行结果]

运行结果如图 9-16、图 9-17 所示。在图 9-17 中信号被清晰地分解成高频细节部分和低频近似部分。通过分解，原信号中的特征就可以轻易地发掘出来。

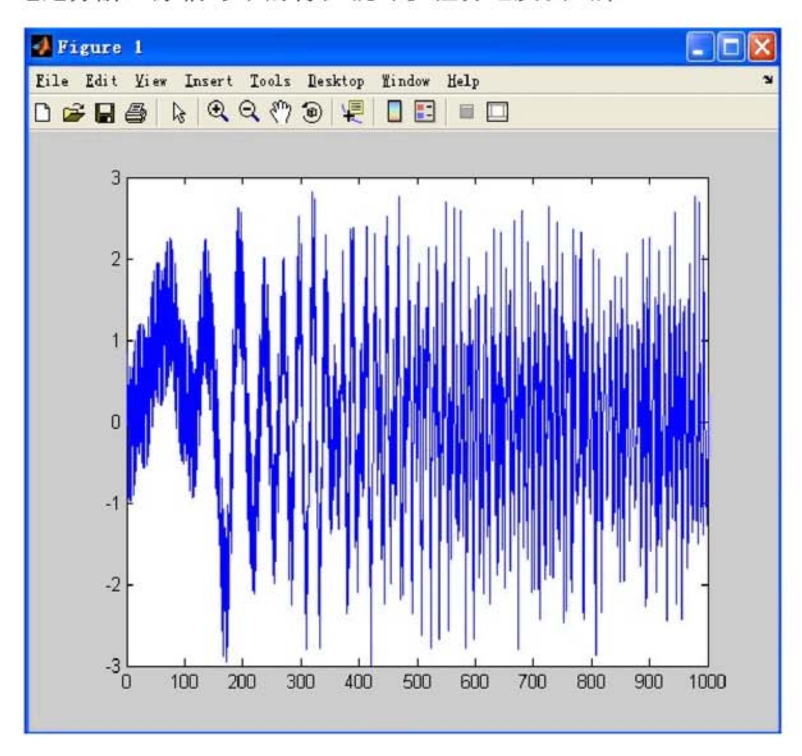


图 9-16　mishmash 的信号曲线

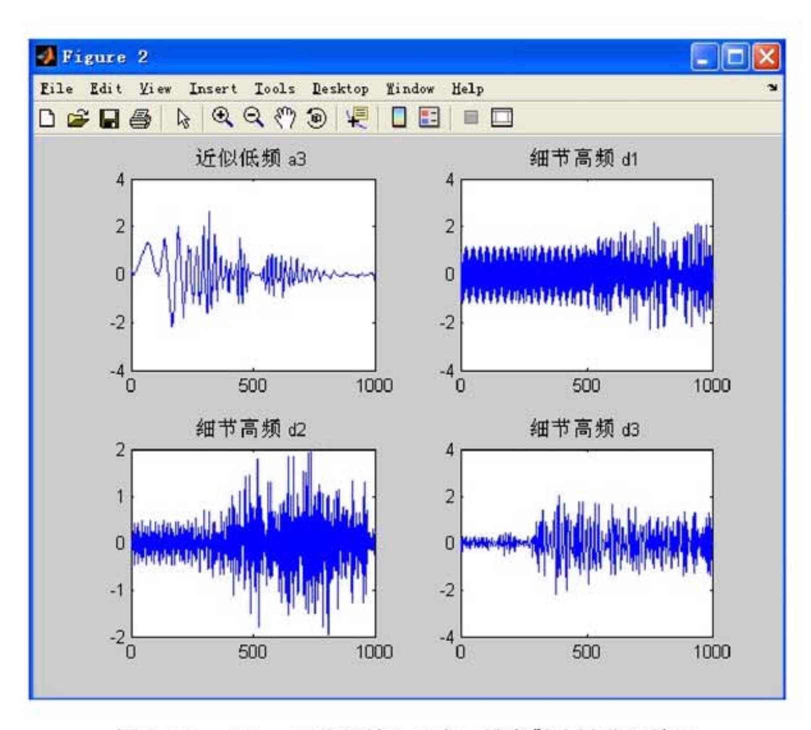


图 9-17　mishmash 信号的 3 尺度一维离散小波分解结果

习题 9

1．简述 MATLAB 7.0 中工具箱的使用方法。

2．试用 MATLAB 求函数 $f(X)=e^{x_1}(4x_1^2+2x_2^2+4x_1x_2+2x_2+1)$ 的极值.

3．试用 MATLAB 求函数 $f(X)=e^{x_1}(4x_1^2+2x_2^2+4x_1x_2+2x_2+1)$ 在限定条件下的极值，限定条件为 $1.5+x_1x_2-x_1-x_2\le 0$，$-x_1x_2-10\le 0$。

4．试用 MATLAB 求函数 $f(X)=e^{x_1}(4x_1^2+2x_2^2+4x_1x_2+2x_2+1)$ 在限定条件下的极值，条件为 $1.5+x_1x_2-x_1-x_2\le 0$，$-x_1x_2-10\le 0; x(1)\ge 0, x(2)\ge 0$。

5．试用 MATLAB 求函数 $f(X)=e^{x_1}(4x_1^2+2x_2^2+4x_1x_2+2x_2+1)$ 在限定条件下的极值，条件为 $x_1^2+x_2=1$，$-x_1x_2\le 10$。

6．简述 MATLAB 中动量 BP 算法的计算步骤。

7．已知 7 个样本输入参数为 p=[1.0 4;1.5 2;1.5 6;2.5 4;2.0 4;3.0 8;3.0 4]，各样本对应的输出参数为 t=[0.1581;0.3261;0.2812;0.4755;0.4107;0.5650;0.6481]，试用 2×8×1 动量 BP 网络(输入层、隐层、输出层单元数分别为 2、8、1)预测新样本 p_1=[2.0 4]的输出结果。

8．举例说明 MATLAB 中的一维连续小波分析。

9．举例说明 MATLAN 中的一维连续复小波分析。

第 10 章　模 型 使 用

本章要点：

- ☑ MATLAB 中的模型建立方法（包括启动建模、复制模块、增加信号线、确定模型参数与仿真方法等）;
- ☑ MATLAB 中模型的打开与修改;
- ☑ 模型使用实例。

Simulink 是用来对动态系统进行建模、仿真和分析的软件包，它支持连续、离散或两者混合的线性和非线性系统，也支持具有多种采样速率的多速率系统。

Simulink 和 MATLAB 集成在一起，不能独立运行，只能在 MATLAB 环境中运行。同时，MATLAB 自身所带的所有应用工具箱，也适用于 Simulink 环境。

10.1　建立模型

10.1.1　启动 Simulink

在 MATLAB 环境中启动 Simulink 的方法如下：

- 在 MATLAB 7.0 平台环境中单击工具按钮 ;
- 在命令窗口中输入“simulink”命令。

Simulink 启动后首先出现的是 Simulink 的库浏览器，如图 10-1 所示。

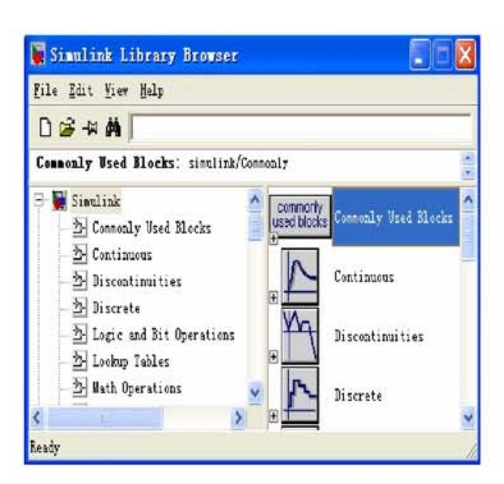


图 10-1　Simulink 的库浏览器

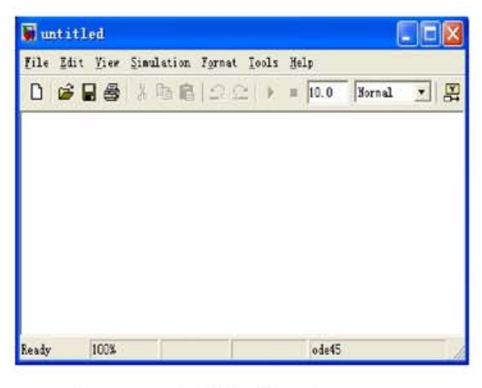
在库浏览器中单击工具条上的图标 ，即可创建一个新的模型窗口，如图 10-2 所示。

Simulink 的工作环境主要由库浏览器与模型窗口组成，前者为用户提供了展示 Simulink 标准模块和专业工具箱的界面，后者是用户创建模型方框图的主要区域。

10.1.2　复制模块

创建模型的第一步是确定模型中包含哪些模块，然后使用库浏览器，在库浏览器中查找到所需要的模块（模块名称不分大小写），将需要的模块从模块库中复制到模型中。

例如，图 10-3 为简单模型方框图，其基本功能是对输入的正弦信号进行积分，然后将积分后的信号连同正弦信号本身送到示波器中显示出来。在这个例子当中，模型包含四个模块，分别是正弦波模块（Sine Wave）、积分模块（Integrator）、示波器模块（Scope）和组合模块（Mux）。涉及的模块库分别是 Sources 库、Continuous 库、Sinks 库和 Signal Routing 库。



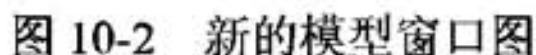


图 10-2　新的模型窗口图

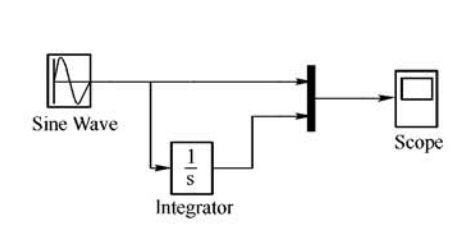


图 10-3　简单模型的方框图

模块的复制可以在模型窗口、库窗口、模型窗口和库窗口之间实现。不同模型窗口和库窗口之间模块复制的方法是：在一窗口中选中模块，按下鼠标左键，将它拖到另一模型窗口，释放鼠标。相同模型窗口内的模块复制方法是：在一窗口中选中模块，按下鼠标右键，拖动鼠标到合适的地方，然后释放鼠标。

复制后所得模块具有和源模块相同的属性，并且在不同模型窗口内时二者同名。如果相同模块在同一模型窗口，Simulink 将在相同模块的名字后面加上相应的数字加以区分。

10.1.3　增加信号线

信号是 Simulink 在仿真过程中模块输入输出的数据流。信号线是指模块间的连接线。各个模块间的信号通过信号线携带与传播，信号线可以把一个模块输出与另一个模块输入连接起来，也可以把其他信号线与一个模块输入连接起来。每一个输出端口可以有任意条信号线，而每一个输入端口却只能有一条信号线（Mux 模块可把几个标量信号线合并成一个向量信号线）。

Simulink 自动在模块周围布线，而不是把信号线从模块中间穿过。不过，如果用户在画信号线或线段的同时，按下了 Shift 键，那么 Simulink 就根据用户的要求画信号线，而不再遵循这一规则。

（1）要把一个模块的输出与另一个模块的输入连接起来，可采用下面的步骤。

- 把鼠标指针移到第一个模块靠近端口的任何位置，光标将变成十字形。
- 按下鼠标，拖动鼠标指针到第二个模块输入端口的任何位置。如果用户把鼠标指针移到了第二个模块的里边，则把信号线连到第二个模块第一个未曾占有的输入端口。如果想把信号线连到指定端口，就必须在释放鼠标按钮以前把鼠标定位到那个输出端口。
- 释放鼠标按钮，Simulink 用一个带箭头的实线信号线代替端口的符号，用来表示信号

的流向。用户既可以把信号线从一个模块输出端口连到另一个模块输入端口，也可以反方向连接。在这两种情况下，箭头都出现在输入端口，而且信号的流向不会因为信号线的方向不同而发生改变。

（2）用户可以在一条已有信号线上引出另一条信号线，这两条信号线将传送相同信号给各自对象。要从某一根信号线上引出另一根信号线，可采用下面的步骤。

- 把鼠标指针移到这根信号线上的某个位置，这个位置就是引出新信号线的起始位置。
- 在按下 Ctrl 键的同时，按下鼠标按钮。
- 拖动鼠标到目标端口，然后释放鼠标按钮和 Ctrl 键，那么 Simulink 就在起始位置和目标端口之间创建了一条新信号线。

10.1.4 确定模型参数

几乎所有的模块都有可以进行参数设置的对话框。在模型窗口中选中一个模块，用鼠标双击该模块，Simulink 将打开模块基本属性对话框。图 10-4 为 Sine Wave 模块属性设置对话框，其各项含义如下。

- Amplitude：输出信号的幅值，默认值为 1。
- Frequency：信号的频率，默认值为 1 弧度/秒。
- Phase：信号的相位移，默认值为 0。
- Sample time：采样时间，默认值为 0，表示工作于连续模式，>0 表示工作于离散模式，-1 表示模块工作模式与接受信号模块相同。
- Interpret vector parameters as 1-D：将向量参数以一维形式表示。

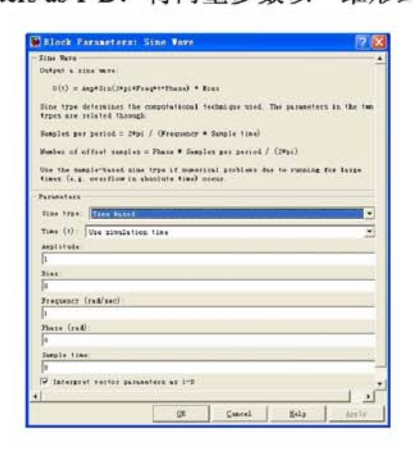


图 10-4 Sine wave 模块属性设置对话框

在模型窗口中选择【Simulation：Configuration parameters】菜单，可进行仿真参数设置。

1．求解器的设置

求解器属性对话框如图 10-5 所示，其功能是完成仿真起止时间、求解器类型及输出选项的设置。

（1）Simulation time（仿真时间设置）

用户可以修改仿真的开始和结束时间，默认时仿真从 0.0 秒开始，在 10.0 秒处结束。注意：这里的仿真时间与实际时钟时间是不同的，后者是仿真计算所花费的实际时间，它的长短与许多因素有关，如模型的复杂程度、仿真的最大步长及计算机的时钟频率。

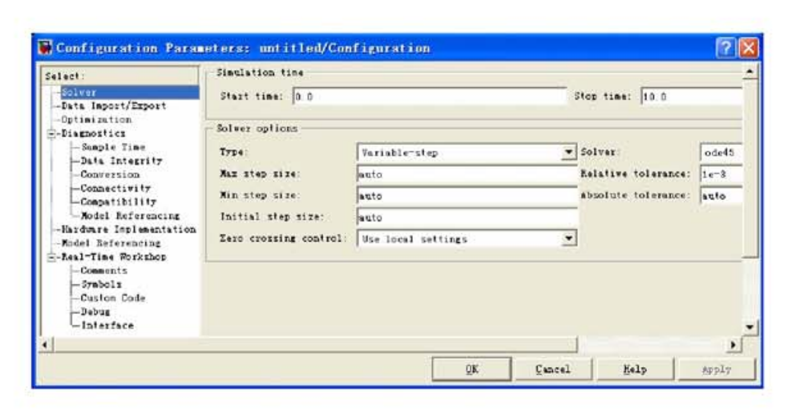


图 10-5　求解器设置对话框

（2）Solvers options（求解器选项设置）

Simulink 模型的仿真一般需要采用微分方程或微分方程组的数值解法。Simulink 为用户提供了各种不同的求解算法，这些算法的使用范围和计算速率各不相同，为了保持仿真的高效性，用户应该根据仿真模型的特点，选择最合适的求解算法。另外，用户还可以在可变步长或固定步长之间选择。可变步长（Variable-step）提供了误差控制机制，而固定步长不论每步仿真的误差有多少，总是采用相同的仿真步长。

在 Simulink 中，可变步长类（Variable-step）包括以下求解方法。

- ode45 为 4 阶或 5 阶 Runge-Kutta 算法，属于一步求解法，即计算当前值 $y(t_n)$只需要前一步的结果 $y(t_{n-1})$，是大多数问题的首选求解算法。
- ode23 为 4 阶或 5 阶 Runge-Kutta 算法，属于一步求解法，在较大的容许误差和中度刚性系统模型下比 ode45 方法更加有效。
- ode113 是变阶的 Adams-Bashforth-Moulton PECE 方法，属于多步预测算法，也就是说计算当前值需要 $y(t_n)$前几步的结果。
- ode15s 为数值微分公式（numerical differentiation formula）的变阶算法，属于多步预测算法，如果系统刚度很大或 ode45 求解效率不高，可以采用这种算法。
- ode23s 为改进二阶公式的算法，属于一步求解算法，在较大容许误差情况下，比 ode15s 更加有效。因此，如果在求解某些刚性系统时 ode15s 计算效果不佳，可改用此法。
- ode23t 一般用来解决中等刚性系统的求解。
- ode23tb 在较大容许误差情况下可能比 ode15s 方法有效。
- discrete（variable-step）当前系统中没有连续状态变量时可选择该方法。

在 Simulink 中固定步长类（Fixed-step）包括以下求解方法。

- ode5 采用固定步长的 ode45 方法。
- ode4 采用 4 阶 Runge-Kutta 算法。
- ode3 采用固定步长的 ode23 方法。
- ode2 采用改进的 Euler 公式。
- ode1 即 Euler 方法。

固定步长离散系统求解算法（discrete_fixed-step），尤其适用于不存在状态变量的系统。

在变步长方式中，Simulink 中的最大步长（Max step size）一般设置成自动（auto）方式，这时最大步长约是仿真时间的 1/50；如果仿真时间很长，也可以设为其他值（对于周期系统

可以定为周期的 1/4）。如果需要了解某段时间内系统动态特性的细节，可将该值调小些。Simulink 中的初始步长（Initial step size）默认为自动（auto）方式，这时系统会根据初始状态导数的大小计算初始步长。如果初始步长选择过大，可能会忽略系统启动时的某些性状。这里的初始步长并不是实际计算步长，而是系统第一次选择的步长，如果该步长计算结果超出了系统容许误差，系统将自动减少步长并重新计算。

在 Simulink 中，对于离散系统或混合系统（Tasking mode for periodic sample times），如果选用固定步长方式进行计算，有如下三个工作方式进行选择。

- 单任务方式：主要用于单任务系统，运行中不检查模块间是否存在速率突变。
- 多任务方式：用于多速率或混合系统，运行时检查各模块的工作速率有无冲突。
- 自动方式：固定步长算法的默认工作方式，自动进行系统检测（如果整个系统采用同一采样速率则以单任务方式运行，否则以多任务方式运行）。

Simulink 中的零点穿越控制（Zero-crossing control）使 Simulink 在进行变步长积分计算时能够进行零点穿越检测。对于大多数模型，让求解器采取更大的时间步长能够加快仿真速度，但如果模型动态变化比较大，应将该项设置为“Disable all”，此时虽然能够提高仿真速度，却降低了仿真精度。

Simulink 中的容许误差（Relative tolerance 和 Absolute tolerance）起到控制计算精度的作用。Simulink 中的微分方程求解器，使用标准局部误差控制算法监测每一步仿真计算误差。每个仿真步骤中，求解器计算所有的状态值和误差大小、并同容许误差进行比较，如果计算误差超过了容许误差限制，则取消本次仿真次骤，并减少仿真步长重新计算。容许误差由相对误差（rtol）和绝对误差（atol）共同决定，即 $e_i \leqslant \max(\mathrm{rtol} \times |x_i|, \mathrm{atol}_i)$。当 x_i本身绝对值较大时，计算精度主要由相对误差控制；当 x_i本身绝对值较小时，计算精度主要由绝对误差进行控制。一般情况下，用户只需指定容许误差为默认值（此时绝对误差初始值为 10^{-6}），之后绝对误差将随着仿真过程进行调整。如果用户要人为指定容许误差值，需要多次仿真才能最终确定合适数值；如果需要为不同状态分别指定不同容许误差，可以在相应的积分模块中进行设置。

2．数据输入输出设置

使用 Simulink 进行动态系统仿真时，用户可以直接将仿真结果输出到 MATLAB 基本工作空间中、也可以在仿真启动时从基本工作空间中载入模型的初始状态，这些数据输入输出的设置都可在工作空间属性对话框中完成，如图 10-6 所示。

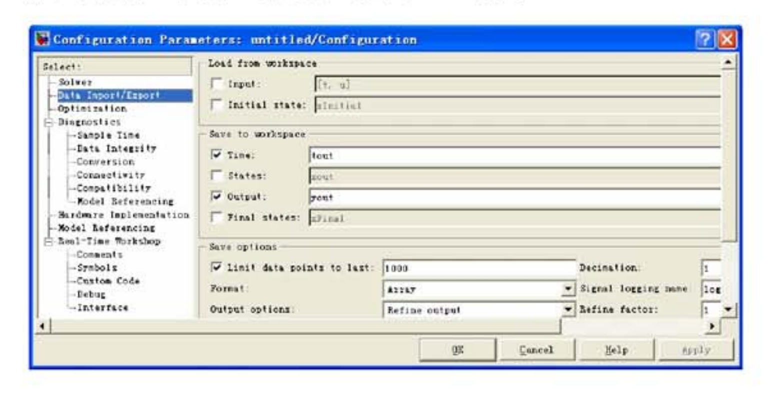


图 10-6 数据输入输出的设置

工作空间属性对话框中各参数的含义如下。

（1）Load from workspace（从基本工作空间中输入数据）

Simulink 在运行过程中可以将基本工作空间的数据载入到仿真模型当中，为此，用户需要选择【Load from workspace】的【Input】，并在相应【Input】文本框中输入需要的数据变量名（例如[t,u]），向量第一列为仿真时间，第二至第 n 列分别对应于模型的第一至第 n 个输入。Simulink 仿真过程中，每行还要输入数据进行插补或跳转以满足仿真步长的需要。选择初始状态【Initial state】，模型将从基本工作空间中获取模型所有内部状态变量初始值，不论模块本身是否设置了初始值，该文本框中应输入基本空间中已经存在的变量，变量的次序应与模块中各个状态的次序一致。

（2）Save to workspace（保存到工作空间中）

- Time 文本框：选择该项，模型将把时间变量以指定的变量名输出到基本工作空间中，其默认名为“tout”。
- States 文本框：选择该项，模型将把其所有状态变量以指定的变量名输出到基本工作空间中，其默认名为“xout”。
- Output 文本框：如果模型窗口中使用输出模块 Out，必须选择该文本框，并填写基本工作空间中的输出数据变量名。数据的存放方式与输入情况相同。
- Final state 文本框：选择该文本框，模型将最终状态值输出到基本工作空间，如果最终状态向量在该模型的新一轮仿真中又被用作初值，那么这新一轮仿真是前一轮仿真的继续。

（3）Save options（变量存放选项）

- Limit rows to last 文本框：该选项可以设定保存变量的数据长度（默认值为 1000），如果输出数据超过设定值，则最早的历史数据将被“冲掉”。
- Decimation 文本框：设置解点保存频率，如果取值为 n、则每隔 n-1 个点保存一个解点，默认值为 1。
- Format 文本框：该选项用于设置保存数据的格式。

（4）Output Options（输出选项）

该选项让用户控制仿真产生输出的数目，用户可以在弹出菜单中选择下列选项之一。

- Refine output：该选项在仿真结果太差时提供更多的输出点，该参数是两个仿真步骤之间额外输出点的个数（该默认值为 1）。例如，对于调整因子为 2 的情况，仿真时将在相邻两步仿真中间时刻额外进行一次计算。采用增大调整因子的方法，可以使仿真曲线更加光滑。额外输出通过连续插值完成、仿真步长并不发生改变，仿真速度要比采用减少仿真步长的方法快的多。仿真步长过大会使系统仿真曲线不光滑，可通过增大调整因子获得更好的仿真结果。
- Produce additional output：该选项可让用户直接指定需要增加的额外输出时间，可以在相应时间域中输入具体时间向量，额外增加的输出通过连续插值实现。值得注意的是，该选项会根据额外输出调整仿真步长。
- Produce specified output only：该选项只输出指定时刻仿真结果，通过改变仿真步长来适应指定的输出时刻。

3．优化属性的设置

优化属性对话框用于（如图 10-7 所示）设置某些高级选项，这些选项可以影响 Simulink

仿真的环境，下面介绍各项含义。

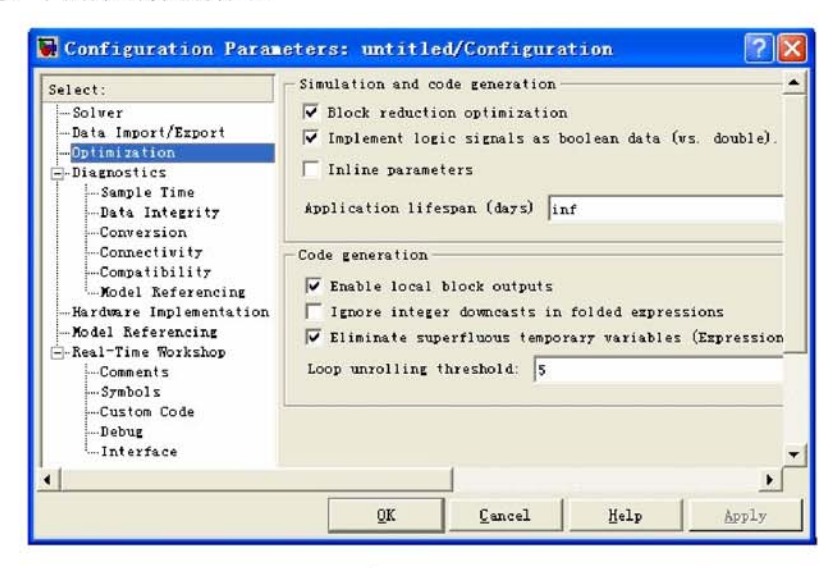


图 10-7 高级属性设置对话框

（1）Simulation and code generation（产生仿真和代码）

Inline parameters：默认情况下，模块许多参数可以在仿真过程中实时调整。如果选择该选项，所有模块参数将不能实时调整（用户专门指定除外），Simulink 将这些不能实时调整的参数视为常数，从而提高仿真计算速度。如果勾选该项，但又要指定某些参数保持实时调整的性质，可以在弹出的模型参数配置（Model parameter configuration）对话框（如图 10-8 所示）中进行设置。方法是单击【Configure】按钮，下面几种参数在仿真过程中实时改变：MATLAB 基本工作空间中定义的变量，在模型参数配置对话框中设置为全局（global）变量的参数。Simulink 为加快仿真将常值信号移到仿真循环之外。

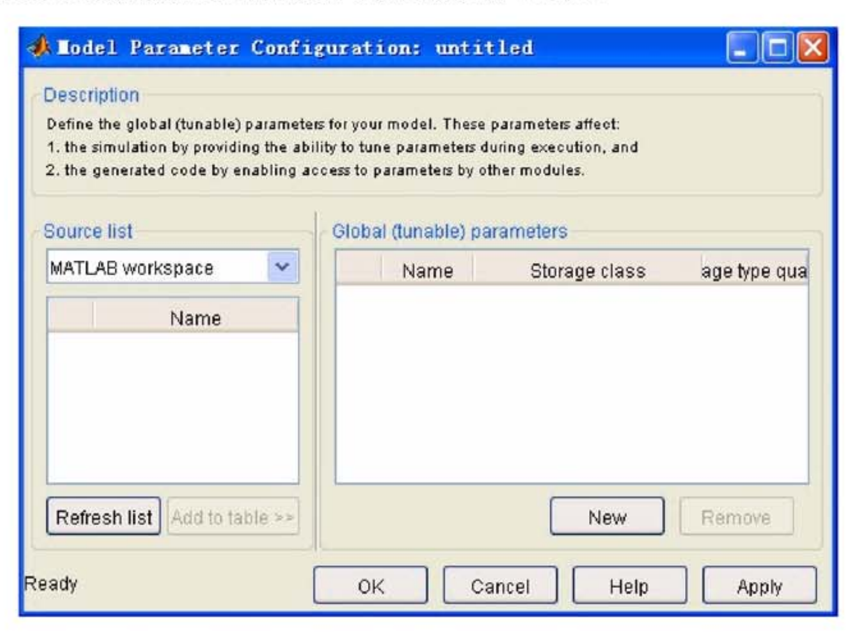


图 10-8 模型参数配置对话框

如果要改变上述参数值，可以修改相应工作空间中的变量值，然后在模型窗口中选择“Edit：Update Diagram”菜单来更新模型方框图。

Block reduction optimization：选择该选项，Simulink将尽量用一个综合模块替代一组模块值，从而加快模型的执行。

Implement logic signals as boolean data：选择该选项，Simulink将能接受布尔类型信号的模块只接受布尔信号，如果不选该项（off），则模块不仅接受布尔信号、同时也能够接受双精度型信号。

Signal storage reuse：如果不选该项（off），则Simulink为每一个模块分配独立的内存缓冲区，这样会极大增加系统内存的消耗，一般只在调试模型、调试CMEX文件形成的S函数时才设置成off状态。

（2）Model parameter configuration：Configure（模型参数配置对话框）

Source list：显示工作空间中变量的列表，可选项包括以下两项。

- MATLAB workspace：列出MATLAB工作空间中具有数值量的所有变量。
- Referenced workspace variables：只列出该模型中引用的变量。

Refresh list：如果在参数配置之后修改了工作区间中的变量，则单击该按钮可以更新变量列表的显示。

Add to table：将【Source list】中选中的变量添加到右边的可调参数列表中。

New：定义一个新的变量，并且将它添加到可调参数列表中，注意该按钮不会在MATLAB工作空间中创建相应变量，用户必须手动创建。

Storage class、Storage type qualifier用于模型的代码生成。

4．诊断页的设置

用户在仿真过程中常常会遇到各种各样的错误或报警消息，错误消息会中止仿真过程，而警告消息则不会。用户可以通过诊断属性页的适当设置，来定义是否需要显示相应的错误或报警消息，诊断属性页如图10-9所示。

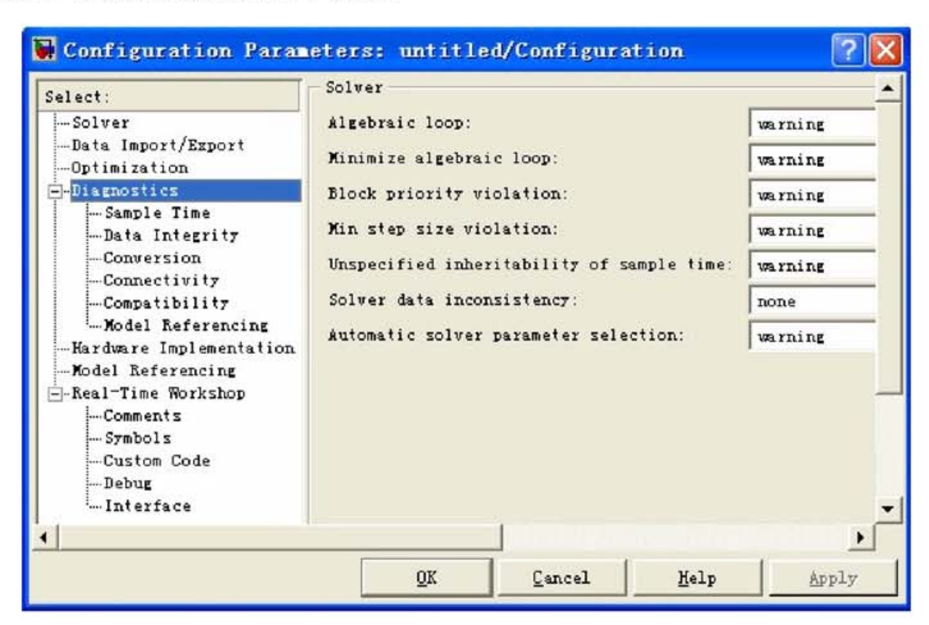


图10-9　诊断页的设置对话框

Simulink 通过内部一致性检测，专门用来调试用户定制模块的编程的正确性，对于由Simulink 标准模块组成的系统一般不需要进行一致性检查。对于各种异常，Simulink 通过设置对话框右侧选项，提供了三种处理方式。

- none 表示 Simulink 忽略这种异常。
- warning 表示 Simulink 遇到这种异常时发出相应警告消息，但不中止仿真过程。
- error 表示 Simulink 遇到这种异常时发出相应错误消息，并中止仿真过程。

下面为 5 种常见异常。

- Algebraic loop：代数环的存在将大大影响模型的仿真速度，对于这种异常，一般采用 warning 的处理方式，如果用户可以忍受代数环所导致的仿真速度减缓，则可以忽略这种警告，否则可以采取适当措施消除代数环。
- Min step size violation：出现这种异常表明，微分方程求解器为达到指定的精度要求而需要更小的计算步长，但这又是求解器不允许的，解决的方法是采用更高阶的求解算法。对于这种异常通常采用 error 或 warning 方式。
- Unconnected block input ports：出现这一异常表明，模型当中存在未被使用的输入端，可以将该输入端接到 Groud 模块的输出端来消除这种异常。处理这种异常一般采用 error 或 warning 方式。
- Unconnected block output ports：出现该异常表明，模型当中存在未被使用的输出端，这种异常对仿真过程无大的影响，只要将该输出端接到 Terminator 模块就可消除这种异常。对于这种异常一般选 error 或 warning 方式。
- Unconnected line：这种异常的出现表明，模型中存在一端未被使用的信号线，这往往是建模疏忽造成的，因此这种异常可选 error 方式进行处理。

10.1.5 仿真

1．仿真过程的启动

完成仿真参数配置后，在模型窗口中选择【Simulation】→【Start】菜单项，仿真开始。而系统发出哗的声音则提示用户仿真结束。

由于模型复杂程度和仿真时间跨度大小不同，每个模型的实际仿真时间不同、不同机器对同一模型进行仿真所需时间也不相同。用户可以在仿真过程中单击【Simulation】→【Stop】菜单项，人为中止模型的仿真。如果模型中包含向数据文件或工作空间输出结果模块或在仿真配置中进行了相关设置，则仿真过程结束会将结果写入数据文件或工作空间中。

2．仿真过程的诊断

如果仿真过程中出现错误，仿真一般会自动停止，并弹出一个仿真诊断对话框来显示错误的相关信息，该对话框分为上下两个部分，上面部分是每个错误消息的基本情况、下面部分是对相应错误的详细描述。仿真诊断对话框将显示如下的错误信息。

- Message：错误类型，诸如模块错误或警告等。
- Source：发生错误的模块名称。
- Fullpath：导致错误的对象的完整路径。
- Summary：错误的简单说明。
- Reported by：报告错误的组件，如是 Simulink、Stateflow 还是 Real-Time Workshop 检测到该错误。

Simulink 除了弹出诊断对话框显示错误信息外，必要时还会弹出模型方框图，并采用高亮方式显示引发该项错误的相关模块，用户可在诊断对话框中双击相关错误弹出的方框图。

3．仿真过程的模式

仿真过程有正常模式（Normal）和加速模式（Accelerator）两种。在模型窗口中，【Simulation】菜单中的【Accelerator】菜单项提供了仿真加速器功能，使得仿真速度可能有本质的提高。

用户在加速模式下启动仿真后，加速器会自动生成 C 语言代码，并放置在 modelname_accel_rtw 的子目录下（modelname 为具体的模型名称），然后将生成的代码编译成可执行文件 mex，mex 文件则放在 MATLAB 当前工作目录下，最后运行编译后的模型。

如果用户改变了模型的结构，例如增加或删除模型中的模块，加速器会自动修改对应的 C 语言代码，并重新进行编译。下面的操作将会影响模型的结构。

- 改变积分的算法。
- 增减模块或改变模块之间的连接。
- 改变模块输入输出端口的数目。
- 改变模型当中状态变量的数目。
- 改变 Trigonometric Function 模块中的函数。
- 改变 Sum 模块中所用的符号。
- 加入 Target Language Compiler™（TLC）文件来产生某个内建的 *S* 函数。

在模型仿真过程中，如果用户对模型的修改影响到模型的结构，Simulink 将忽略这种改变，并且出现警告信息。用户如果想要修改生效，必须停止仿真过程，待修改完成后，重新开始。

需要注意的是，加速器不会显示仿真过程本身产生的警告信息，如被零除和数据溢出，这点与前面所讲的情况不大一样。加速模式下，不允许采用单步跟踪的调试步骤。

4．仿真结果的观察

仿真进行当中，用户一般需要随时绘制仿真结果的曲线，以观察信号的实时变化，在模型当中使用示波器（Scope 模块）是其中最为简单和常用的方式，在模型窗口选中示波器（Scope 模块）后，鼠标左键双击 Scope 模块，将显示出示波器窗口，如图 10-10 所示。

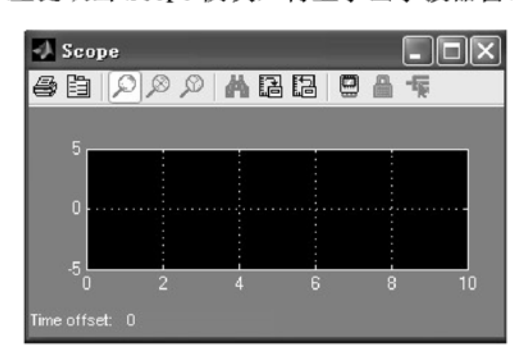


图 10-10　Scope 模块的显示窗口

在示波器窗口的显示范围内单击鼠标右键，将弹出一个上下文菜单，选择【Axes properties】菜单项，将弹出纵坐标设置对话框，如图 10-11 所示。在相应的文本框中输入所希望的纵坐标上下限，可以调整示波器实际纵坐标显示的范围。

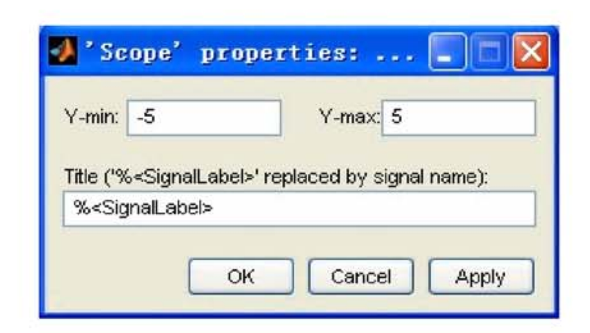


图 10-11 纵坐标设置对话框

在示波器窗口用鼠标左键单击工具按钮，弹出 Scope 模块的参数设置窗口，如图 10-12 所示。



图 10-12 Scope 模块的参数设置窗口

该参数设置窗口的各部分设置如下。

- Number of axes：默认值为 1，此时 Scope 模块只有一个输入口，示波器模块只有一个信号显示区。如果输入值为 2，Scope 模块有两个输入口，示波器窗口右两个信号显示区，依此类推。
- Time range：默认值为 10，表示显示数据的区间在[0，10]内，如果数据的实际范围超出设定的区间，则超出的部分不显示。
- Sampling：包含两个下拉菜单。Decimation 表示显示频度（默认值为 1）。如果取 n，则每隔 n-1 个数据点给予显示。Sampling time 表示信号的采样时间步长(默认值为 0)，取-1 表示显示方式取决于实际输入信号，取大于零的整数、则表示显示离散信号的时间间隔。
- Floating scope：设置示波器是否以浮动窗口的形式出现。
- Limit rows to last：设定缓冲区接受数据长度（默认为选中状态、数据长度为 5000)，如果数据长度超过设定值，则最早的历史数据将被冲掉。
- Save data to workspace：将示波器缓冲区中保存的数据送入 MATLAB 基本工作空间中，变量名可以修改。

10.1.6 保存模型与打印结果

用户可以在模型窗口中选择【Save】或【Save as】命令来保存模型，文件扩展名为.mdl。

Simulink打印模型的方法可以采用菜单方式，选择模型窗口中的【File】→【Print】菜单项，将出现一个打印对话框。它与Windows标准打印对话框的区别在于增加了附加选项框，用于设置打印内容，包括以下内容。

- 只打印当前系统。
- 打印当前系统及其上层系统。
- 打印当前系统及其下层系统
- 打印模型中所有系统，其中可以选择打印封装子系统和库模块的内容。
- 打印每个方框图的框架。

当用户选择打印当前系统时，打印对话框如图10-13所示，而如果用户选择打印全部模型时，打印对话框则会增加两个选项。如图10-14所示。

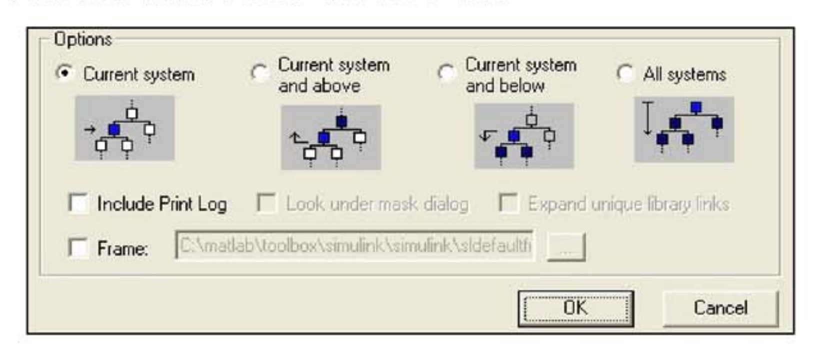


图10-13 打印当前系统的打印对话框

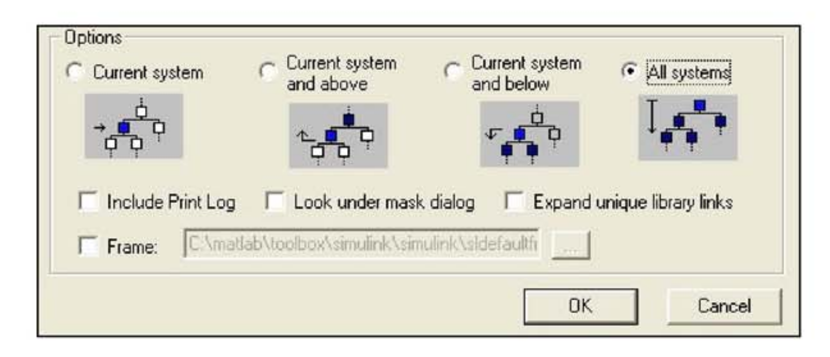


图10-14 打印整个模型的打印对话框

10.2 打开与修改模型

10.2.1 打开模型

模型文件是指在Simulink环境中记录模型中模块类型、模块位置及各个模块相关参数等信息的文件，其文件扩展名为.mdl。打开已存在的模型文件，有如下不同的方法。

- 在 MATLAB 命令窗口，鼠标左键单击工具按钮，打开需要的模型文件，此时不出现 Simulink 的库浏览器。
- 在库浏览器或模型窗口，鼠标左键单击工具按钮，打开需要的模型文件。

10.2.2 添加模块注释

在模型窗口中，书写注释的目的是帮助用户更好地理解模型，添加模块注释包括以下几方面：

（1）注释文本的创建

在注释区中心位置双击鼠标左键，出现编辑框，在框中输入所需的文本后，单击编辑框以外的区域，完成注释创建。

（2）注释位置的移动

在注释文字处单击鼠标左键，待出现编辑框后按下鼠标左键，就可把该编辑框连同其中的内容拖到希望的位置。

（3）注释文字的字体控制

单击注释编辑框，再选择【Format】→【Font】菜单项，弹出标准的 Windows 字体对话框，可选择字体及文字大小。

10.2.3 修改模块

（1）模块的选定

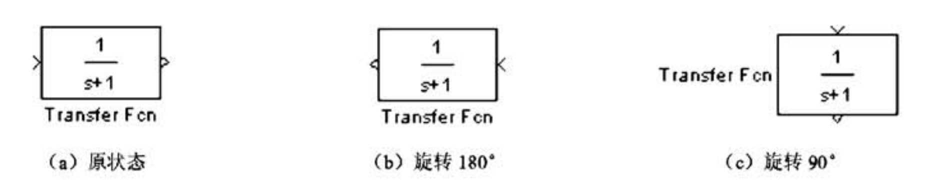
选定单个模块时，用鼠标左键指向待选的模块，单击鼠标左键即可；选定多个模块时，可以按下 Shift 键依次选定所需的模块，或者按住鼠标左键，拉出虚线框，将所有待选定的模块包含其中，然后松开鼠标。

（2）模块的移动

选定所需的模块，按住鼠标左键可以快速拖动模块到希望的位置，也可以用键盘上的上、下、左、右四个键慢速移动模块到合适位置。

（3）改变模块的方向

默认情况下，信号总是从模块的左边流进、从模块的右边流出，即输入在左边、输出在右边，如图 10-15（a）所示。单击【Format】→【Flip Block】菜单项，可将模块旋转 180°，如图 10-15（b）所示。单击【Format】→【Rotate Block】菜单项，可将模块旋转 90°，如图 10-15（c）所示。



（a）原状态　（b）旋转 180°　（c）旋转 90°

图 10-15　多模块方向改变示意图

10.2.4 修改信号线

模型窗口中，传递信号性质不同、信号线会呈现不同状态，默认情况下，黑色表示连续

信号经过的模块，红色表示离散系统模型的最高采样频率。必要时可以利用模型窗口中【Format】菜单下的相关选项进行宽度和颜色的修改。修改信号线形状和位置有以下几种方法。

- 一般方法：将鼠标指向待移动信号线，按下鼠标左键，拖动鼠标至适当位置、然后释放鼠标。
- 多线引出：按住 Ctrl 键，在需要建立分支的地方用鼠标拉出，或者只按住鼠标的右键从需要建立分支的地方拉出，如图 10-16 所示。
- 分割信号线：将鼠标指针移到信号线上的某一点，按下 Shift 键的同时，按下鼠标左键，然后可拖动鼠标到用户所希望的位置，如图 10-17 所示 。

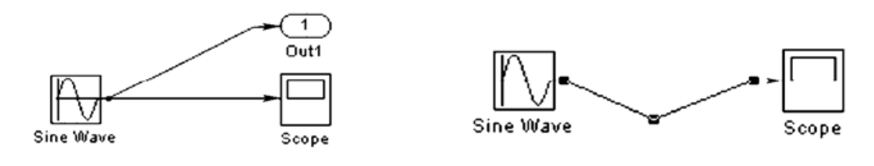


图 10-16 多线输出示意图　　图 10-17 分割信号线示意图

10.2.5 修改模型参数

修改模型参数，主要是修改模块参数，可在模型窗口中选中一个模块，用鼠标双击该模块，Simulink 将打开一个模块的基本属性对话框，在属性设置对话框内对其进行参数设置，修改方法与确定模块参数设置的方法一致，这里不再赘述。

10.2.6 模型分组

在动态仿真过程中，如果被研究的系统比较复杂，那么直接用基本模块构成的的模型就比较庞大，若把整个模型按照实现功能或对应物理器件不同而划分成不同模块，有利于整个系统的概念抽象。模块分组是实现这一目的的最好方法，它包括下面几个步骤。

（1）生成子系统：在模型窗口中，用方框将需要包含进子系统的模块选中，单击【Edit】→【Create subsystem】菜单项，便将选中的模块包装在一个名为 Subsystem 的模块中，如图 10-18（a）和图 10-18（b）所示。

（2）模块名的修改：用鼠标左键双击新生成的子系统中的模块名（Subsystem），将其改为其他名称。

（3）输入输出端口的设置：双击新生成的模块（Subsystem），可以显示模块的基本模型，如图 10-19 所示，该基本模型的输入输出模块可以改为适当的名称。

（4）子系统的保存：单击【File】→【Save】菜单项，将以上创建的子系统进行保存。

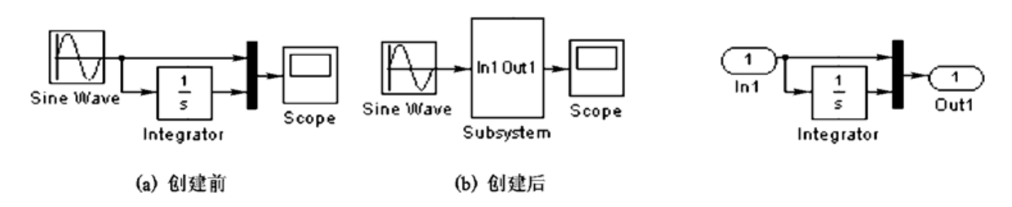


图 10-18 多子系统模型的创建　　图 10-19 Subsystem 模块的基本模型

10.3 应用实例

[例 10-1] 模拟 $Y(S)/X(S)=G(S)/[1+G(S)]$，其中 $G(S)=50/S^2+2S+4$。

（1）首先从 Simulink 的模块库中把需要的模块复制到工作区，如图 10-20 所示。它们分别是Souces模块库里的Step模块(产生一步阶波)、Math模块库里的Sum(求和)、Continuous模块库里的 Transfer Fcn 模块（传递函数）和 Sinks 模块库里的 Scope 模块（示波器）。

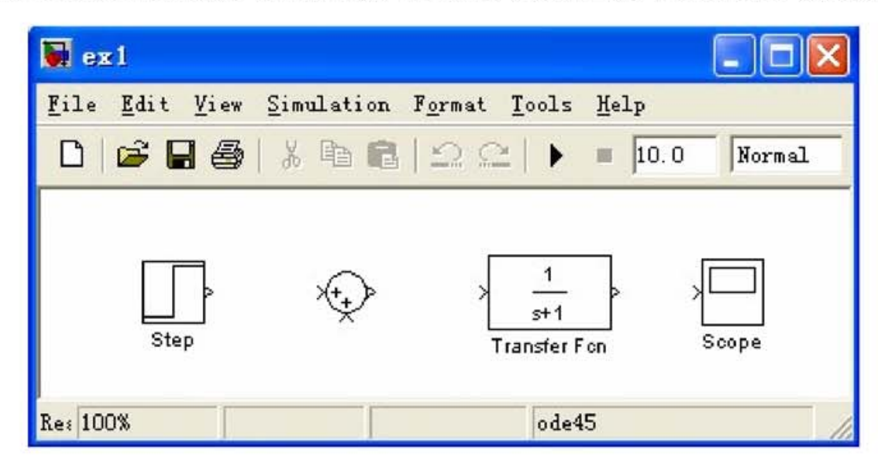


图 10-20 复制模块

（2）设置 Sum 模块参数。双击【Sum】模块，将弹出【Sum】模块对话框，如图 10-21 所示，按照图示的参数进行设置。

（3）设置 Transfer Fcn 模块参数。双击【Transfer Fcn】模块，将弹出【Tranfer Fcn】模块对话框，如图 10-22 所示，按照图示的参数进行设置。

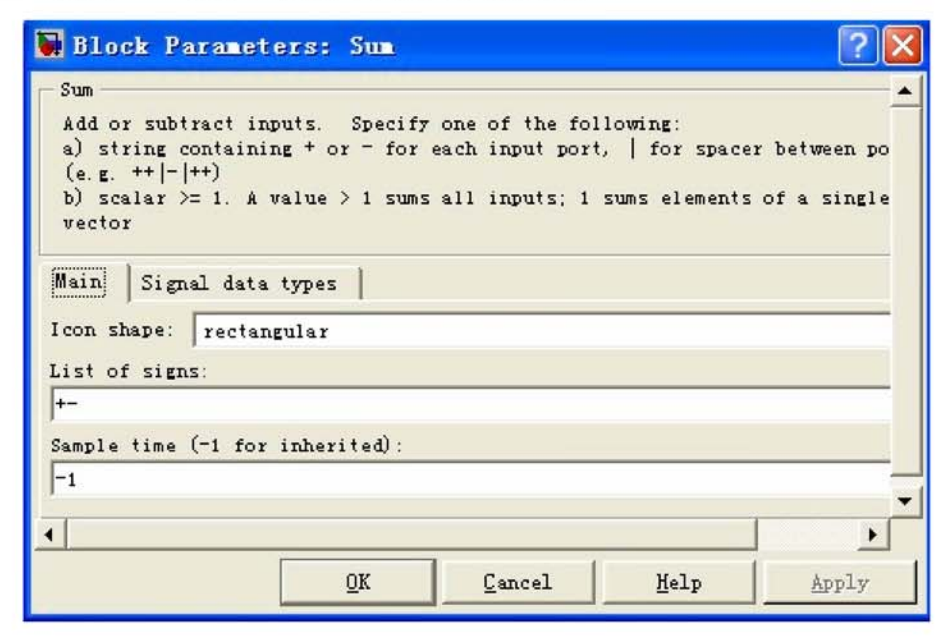


图 10-21 Sum 模块对话框

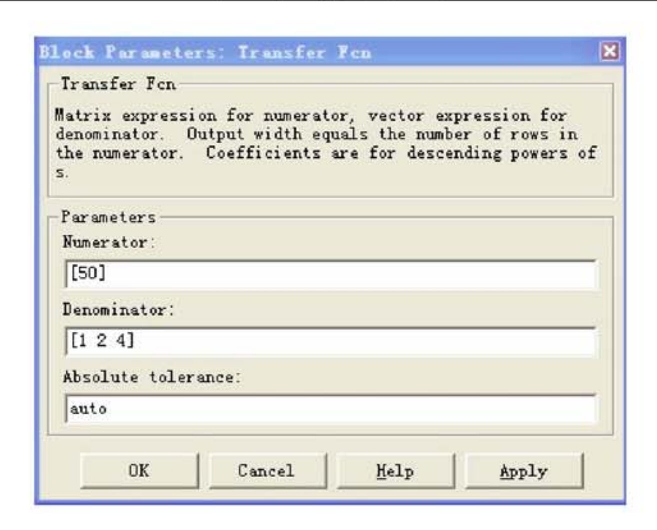


图 10-22　Transfer Fcn 模块对话框

（4）如图 10-23 所示进行连线。

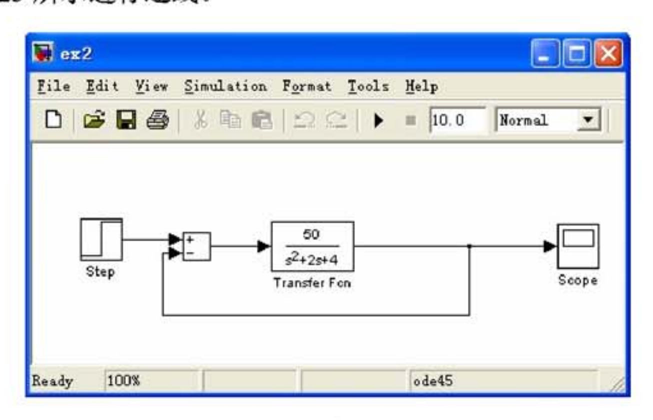


图 10-23　模块连线图

（5）单击【Simulation】→【Start】菜单项，然后双击模块就可看到仿真的结果，如图 10-24 所示。由图 10-24 可以看出，阶跃响应不够理想，在自动控制理论里，可以在系统中加入一个补偿器、然后再模拟结果，如果用户不满意、可根据结果调节有关参数。

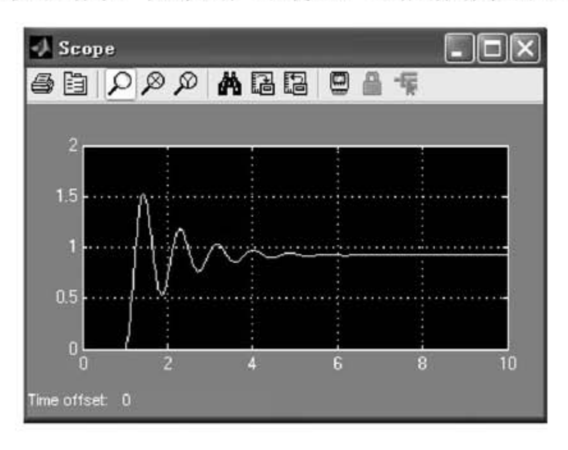


图 10-24　模拟结果图

（6）加上一个补偿器 G_C=（1+0.8S）/（1+0.08S），这个补偿器也是 Transfer Fcn 模块，将其命名为 Compensator，模块连线如图 10-25 所示，模拟结果如图 10-26 所示，这时阶跃响应的结果比前面改善了很多。

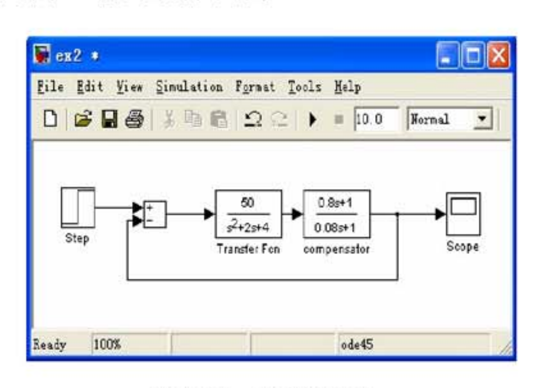


图 10-25 模块连线图

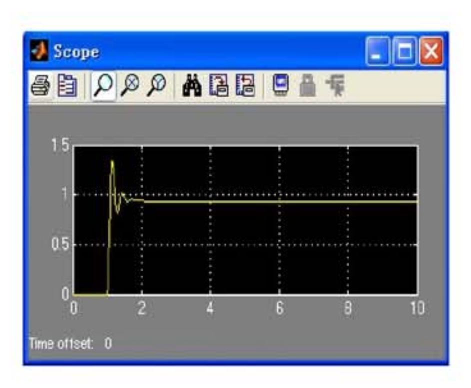


图 10-26 模拟结果图

[例 10-2] 对系统 $y(t)=\begin{cases}5u(t), t\geq 25\\ 2u(t), t\leq 25\end{cases}$ 进行仿真，$u(t)$为系统输入，$y(t)$为系统输出。

（1）建立系统模型。首先根据系统数学描述选择合适的系统模块。本例使用的系统模块有：Souces 模块库里的 Sine Wave 模块（系统输入信号）、Math 模块库里的 Relational Operator 模块（实现系统中的时间逻辑关系）、Souces 模块库里的 Clock 模块（系统运行时间）、Souces 模块库里的 Constant 模块（系统常数值）、Signal Routing 模块库里的 Switch 模块（实现系统输出选择）、Math 模块库里的 Gain 模块（实现系统中的信号增益）和 Sinks 模块库里的 Scope 模块（表示示波器）。编辑完成的模块图如图 10-27 所示。

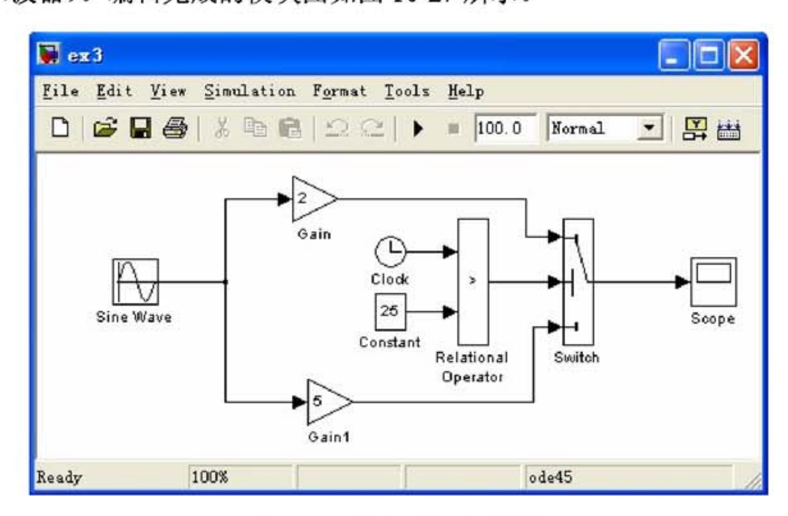


图 10-27 简单系统的建模图

（2）模块参数设置。系统中各模块编辑完成后，需要对各模块参数进行合理设置。本例中的模块参数作如下设置：Sine Wave 模块采用默认参数设置；Relational Operator 模块参数设置为“>”；Clock 模块采用默认参数设置；Switch 模块的 Threshold 设为 0；Gain 模块

的增益值设为 2；Gain1 模块的增益值设为 5。

（3）系统仿真参数配置及仿真分析。Simulink 默认的仿真起始时间是 0s，仿真结束时间为 10s。而本系统当时间大于 25s 时输出才会开始转换，因此需要设置合适的仿真时间，在仿真参数设置对话框的【Solver】选项卡中设置系统合适的仿真时间，可设置仿真起始时间和结束时间分别为是 0s 和 100s。

（4）系统仿真。模块参数和系统仿真参数设置完成后，就可以开始系统仿真（仿真时可先打开 Scope 模块以便实时查看系统的仿真结果）。仿真结束后发现输出曲线非常不平滑，而对系统进行分析可知，系统输出应该为光滑曲线。这是由于没有设置合适的仿真步长而使用默认仿真步长所造成的，因此可在仿真步长进行适当设置后重新仿真。

（5）设置仿真步长。仿真参数的选择对仿真结果有很大影响。此系统的仿真开始时间是 0s，结束时间为 100s，而 Simulink 的最大步长默认值是“（仿真结束时间-仿真开始时间）/ 50”，因此本例仿真步长为 2，这是造成系统仿真输出曲线不光滑的原因所在。在仿真参数设置对话框的【Solver】选项卡中把【Max Step Size】设置为 0.1（即强制仿真步长不超过 0.1），重新启动仿真，示波器模块中获得光滑曲线，如图 10-28 所示。

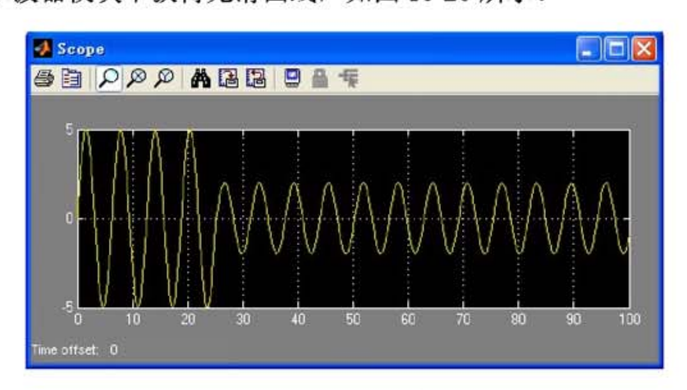


图 10-28　光滑的仿真曲线

另外，MATLAB 为用户提供了许多演示实例（Demos），通过学习这些 Demos，可以获得很多有用的知识。在 MATLAB 命令窗口中，单击【Help】→【Demos】菜单项，弹出【Help】窗口，如图 10-29 所示，在【Simulink】选项中可找到需要的 Demos 模型。

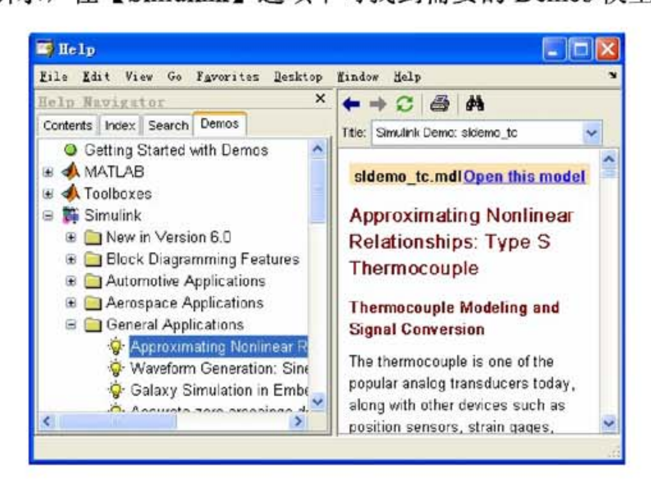


图 10-29　运行 Demos 的【Help】窗口

习题 10

1．以一例说明模型及子系统的建立方法。

2．模拟 $Y(S)/X(S)=G(S)/[2+G(S)]$，其中 $G(S)=20/S^2+S+4$。

3．对系统 $y(t)=\begin{cases}5u(t), & t\geq 25\\ 2u(t), & t<25\end{cases}$ 进行仿真，$u(t)$为系统输入，$y(t)$为系统输出。

4．一因果系统可由微分方程 $y'(t)+3y(t)+2y(t)=f(t)$ 描述，输入为 $f(t)=2u(t)$，系统初始状态为零，求取系统响应。

5．模拟二阶系统 $\ddot{x}_n=[k_n(x_{n-1}-x_n)]/m_n+0.9999x=0$，假设系统从静止状态在 $x=1$ 处启动。

6．一离散系统可由差分方程 $\begin{cases}x_1(k+1)=x_1(k)+0.1x_2(k)\\ x_2(k+1)=-0.05\sin(x_1(k))+0.094x_2(k)+f(k)\end{cases}$ 描述，$f(t)$是输入控制信号且 $f(k)=0.75-x_1(k)$，对该系统进行仿真。

7．著名的 Van del Pol 方程可以描述为 $\begin{cases}\dot{x}_1=x_2\\ \dot{x}_2=-\mu(x_1^2-1)x_2-x_1\end{cases}$，如果 $\mu=1$，$x_1(0)=2$，$x_2(0)=2$，试用 Simulink 仿真获得信号 x_1 和 x_2。

第 11 章　编译器与外部接口

本章要点：

☑ MATLAB 编译器 4.0 的特点和使用方法；
☑ MATLAB 语言与 Excel 接口，包括 EXCEL Link 的安装与操作；
☑ MATLAB 语言与 VB 接口，包括 COM 生成器及其 VB 调用方法；
☑ VC 调用 MATLAB 引擎的方法；
☑ VC 编译 MATLAB 的 mex 文件的方法。

MATLAB 强大的计算与可视化功能、简单易用的开放式可扩展环境、面向不同领域而扩展的工具箱使 MATLAB 成为计算机辅助设计与分析、算法研究和应用开发的基本工具和重要平台。由于编译器采用伪编译方式，MATLAB 编写的程序无法脱离其工作环境而独立运行。针对这个问题，Mathworks 公司为 MATLAB 提供了应用程序接口，允许 MATLAB 和其他应用程序进行数据交换，从而其他程序设计语言可以调用 MATLAB 的高效算法。

本章介绍 MATLAB 编译器 4.0 的特点和使用方法、EXCEL Link 的安装与操作、COM 生成器及其 VB 的调用方法、VC 调用 MATLAB 引擎的方法、VC 编译 MATLAB 的 mex 文件的方法。

11.1　MATLAB 编译器 4.0

11.1.1　MATLAB 编译器 4.0 的特点

与以前版本相比，MATLAB 7.0 采用的编译器 4.0 有许多改进，可以生成分发给其他用户的应用程序、库、COM 对象，其主要特点如下：

- 使用 MATLAB 组件运行时(MCR)取代 MATLAB 数学和图形库。
- 只为接口函数生成代码。
- 具有与代码生成和格式化有关的选项，包括了几个新选项，取消了一些打包选项与相关打包文件。
- 只支持 Microsoft Windows 和 Linux 操作系统。
- 不包括 Visual Studio 插件。
- 不会加速程序运行。
- 不支持 mex 文件和 Simulink S 函数，不支持用 loadlibrary 函数载入而生成的库。

11.1.2　MATLAB 编译器的使用

MATLAB 编译器(相应命令为 mcc)可生成独立应用程序、库、COM 对象、Excel 插件，

它根据目标类型生成合适的包装器文件。包装器文件包含编译后的应用程序和可执行对象类型之间的接口，具有下面一些功能。

- 完成包装器指定的初始化和终止运行工作。
- 定义包含路径信息、加密密匙和 MATLAB 组件运行时(MCR)所需其他信息的数组。
- 提供传递接口函数。

MATLAB 编译器生成的组件技术文件(CTF)与最后生成的目标类型(可执行程序或库)是独立的，包装器文件提供了与目标类型的必要接口。

1．环境配置

mcc 命令具有将 m 文件编译生成 exe 文件或 cpp 文件等许多编译功能，使用 mcc 命令之前必须进行环境配置。mbuild 是 MATLAB 提供的常见编译命令，在命令窗口输入命令“mbuild - setup”完成配置，配置时第一个问题可选“n”，编译器可选用“Microsoft Visual C/C++ version 6.0”。mex 的配置将在 11.5 节中讲述。

2．mbuild 命令

使用命令 mbuild 可对已有 C 文件或 C++文件进行编译，编译生成的 obj 文件可直接双击运行。比如，要对文件 good.c 和 good.cpp 进行编译，可在命令窗口分别输入命令“mbuild – c good.c”和“mbuild – c good.cpp”。

3．mcc 命令

使用 mcc 命令可生成独立可执行文件或 C 共享库。比如，根据文件 file1.m 和 file2.m 生成独立可执行文件使用命令“mcc – m file1.m file2.m”，而根据文件 file1.m 和 file2.m 生成 C 共享库使用命令“mcc – l file1.m file2.m”。其他命令可参阅 MATLAB 自带的帮助文档。

4．使用实例

下面结合例子说明独立应用程序的生成方法。

（1）编辑 m 文件

用 MATLAB 程序编辑绘图程序 mytest.m，代码如下。

```
function mytest()                                   % 定义函数
x=[0,1470,2205,2940,3675,4410,5145,5880,6615,7350];
                          %定义变量 x(静载曲线横坐标)为一维数组并赋值
y=[0.00,1.13,1.98,2.95,4.12,5.63,7.26,9.15,11.43,14.23];
                          %定义变量 y(静载曲线纵坐标)为一维数组并赋值
plot(x,y,'*',x,y,'k-')    %绘制曲线，星号显示数据点
grid on;                  %坐标分格线
axis ij;
                          % 定义坐标原点左上角，横坐标值向右增大，纵坐标值向下增大
xlabel('荷载(kN)');        % 定义 X 轴名
ylabel('位移(mm)');                            % 定义 Y 轴名
title('荷载-位移曲线');                          % 定义图名
axis square;                                  % 图形设置为方形
axis on;                                      % 显示坐标轴上标记、格线和单位标志
uiwait(msgbox('静载曲线绘制完毕'));                % 弹出消息对话窗口
close
```

将 mytest.m 文件保存在指定目录（如 E:\matlab_example\mytest），同时，将工作目录选为该目录。在命令窗口执行命令“mcc -m mytest.m”后，将新增文件夹 mytest_mcr 及 mytest_mcc_component_data.c、mytest_main.c、mytest.ctf、mytest.exe 等相关文件。

（2）测试应用程序

运行上述 mytest.exe 文件，弹出运行结果，如图 11-1 所示。

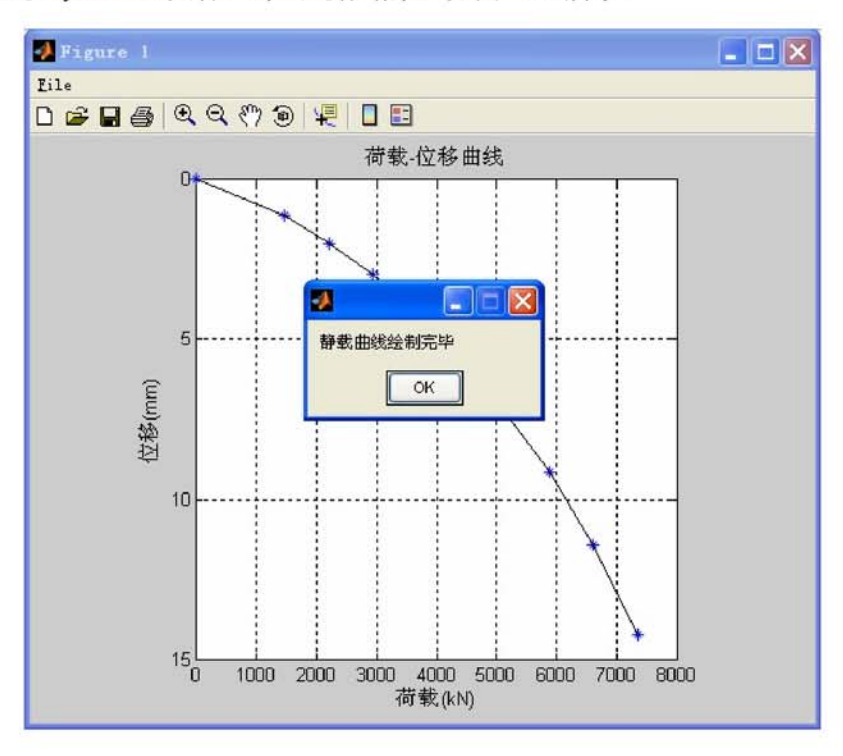


图 11-1　应用程序运行结果

（3）分发应用

MATLAB 编译器生成的独立应用程序可分发到其他机器（目标机器）运行（两台机器的操作系统必须相同），操作步骤如下。

- 执行命令 buildmcr，生成 MATLAB 组件运行时（MCR）库文档，MCRInstaller.zip 将保存在<matlabroot>/toolbox/compiler/deploy/win32 目录下(<matlabroot>为 MATLAB7 安装目录，下同)，该目录下还有文件 MCRInstaller.exe 和 unzip。
- 将下列文件打包并分发到目标机器指定目录：MCRInstaller.zip（Linux 下 MCR 库文档）、MCRInstaller.exe（Windows 下自解压 MCR 库工具）、unzip（Linux 下文件 MCRInstaller.zip 解压工具）、mytest.ctf（组件技术文件）、mytest.exe（应用程序）。

（4）在目标机器上安装和运行

- 将上述打包文件粘贴到目标机器某一位置。
- 目标机器上解压打包文件，运行 MCRInstaller.exe，将 mytest.exe 和 mytest.ctf 复制到应用目录下（如 C:\approot），添加目录<mcr_root>\runtime\win32 到用户系统路径（<mcr_root>为 MCR 安装路径）。
- 运行应用程序。

11.2　MATLAB 与 Excel接口

MATLAB 与 Excel 有两种接口方式：一种是通过 MATLAB 提供的 Excel 生成器，生成

DLL 组件和 VBA 代码，实现 Excel 对 MATLAB 的调用；另一种是利用 MATLAB 提供的 Excel link 插件，直接在 Excel 环境下运行 MATLAB 命令，完成与 MATLAB 的数据传输。下面介绍的是第二种接口方式——Excel link。

Excel Link 是 Microsoft Windows 环境下实现 MATLAB 和 Microsoft Excel 进行链接的插件。通过 MATLAB 与 Excel 链接，用户可以在 Excel 工作环境中利用宏工具及 MATLAB 数据处理和图形处理功能进行相关操作，由 Excel link 进行 MATLAB 和 Excel 工作环境中的数据交换和同步更新。使用 Excel link 时，不必脱离 Excel 环境，可直接在 Excel 工作区或宏操作中调用 MATLAB 函数。

11.2.1 Excel link 的安装和操作

1．系统需求

Excel link 需要的操作系统是 Microsoft Windows XP、Microsoft Windows NT 或 Microsoft Windows 2000，另外还需要 5.1 以上版本的 MATLAB 和 Excel 98、Excel 2000 或 Excel 2002。

2．Excel中注册 Excel link

系统需要在 Windows 环境下先安装 Excel，然后再安装 MATLAB 和 Excel link。安装 Excel link，在 MATLAB 安装组件选择框中选中 Excel link 即可，安装完毕后必须在 Excel 中进行相应设置，具体操作过程如下。

- 启动 Excel。
- 在【工具】菜单中选择【加载宏】选项，打开【加载宏】对话框，单击【浏览】。
- 在弹出的路径选择对话框中，选择“<matlabroot>\toolbox\exlink”下的 Excel 插件“excllink.xla”，单击【确定】按钮。
- 返回【加载宏】对话框，单击【确定】按钮，弹出 MATLAB 运行窗口。
- 稍后，Excel Link 工具条在 Excel 工作窗口左上角出现。工具条包括 “startmatlab”，“putmatrix”，“getmatrix”和“evalstring”4 个工具按钮（见图 11-2），分别为启动 MATLAB、将数据传给 MATLAB、从 MATLAB 提取数据和执行 MATLAB 命令。

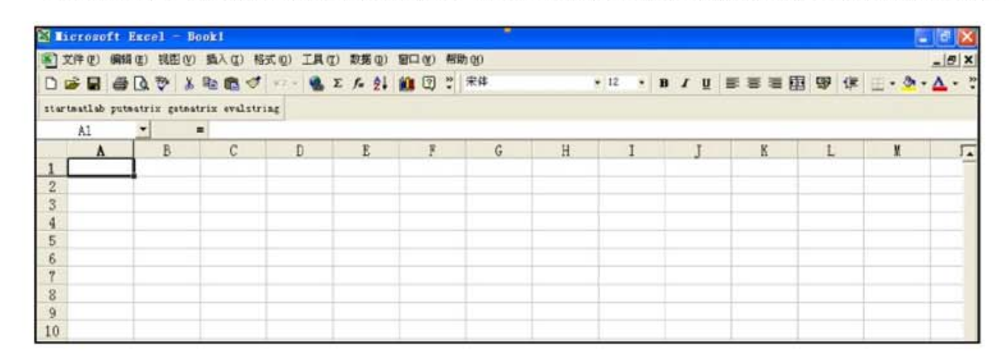


图 11-2　Excel Link 工具条

3．启动 ExcelLink

启动 Excel Link 分自动启动和手动启动两种方式。

（1）自动启动

安装和注册 Excel Link 之后启动 Excel 时，将自动启动 MATLAB 和 Excel Link。

如果用户不想在打开 Excel 时同时启动 MATLAB 和 Excel Link，可在 Excel 数据表单元

格中输入=“MLAutoStart（“no”）”后按 Enter 键，如图 11-3 所示，此函数将更改初始化文件，再次启动该文件时 MATLAB 和 Excel Link 不再自动启动。

图 11-3 A1 单元中输入“=MLAutoStart("no")”

（2）手动启动

➢ 单击【工具】菜单，选择【宏】。

➢ 在弹出 【宏】对话框的【宏名】文本框中输入“matlabinit”，单击【执行】按钮即可。

4．终止 ExcelLink

终止 Excel 时，Excel Link 和 MATLAB 将同时终止。要在 Excel 中终止 MATLAB 和 Excel Link 的运行，可在 Excel 数据表单元格中输入“=MLClose()”并按 Enter 键。重新启动时，可在 Excel 数据表单元格中输入“=MLOpen()”后按 Enter 键或在【宏名】文本框中输入“matlabinit”手动启动。

11.2.2 Excel link 的函数

Excel Link 提供了一系列管理链接、操作数据的函数，可在 Excel 环境中通过宏或工作表单元公式调用这些函数，实现 Excel Link 和 MATLAB 的数据交换与同步更新。

1．链接管理函数

Excel Link 提供了四个链接管理函数来初始化、启动、终止 Excel Link 和 MATLAB。

Matlabinit：初始化 Excel Link 并启动 MATLAB 进程。

MLAutoStart：自动启动 MATLAB 进程。

MLClose：终止 MATLAB 进程。

MLOpen：启动 MATLAB 进程。

可在 Excel 的【工具】菜单中单击【宏】选项或在宏过程中调用 matlabinit 函数，其他链接管理函数可用宏或工作表单元公式进行调用。

2．数据管理函数

Excel Link 提供了 9 个数据管理函数在 Excel 和 MATLAB 之间复制数据、在 Excel 中运行 MATLAB 命令。

Matlabfcn：对给定 Excel 数据运行 MATLAB 命令。

Matlabsub：对给定 Excel 数据运行 MATLAB 命令并指定输出位置。

MLAppendMatrix：将 Excel 工作表中数据创建或添加到 MATLAB 矩阵。

MLDeleteMatrix：删除 MATLAB 矩阵。

MLEvalString：运行 MATLAB 命令。

MLGetMatrix：将 MATLAB 矩阵内容写到 Excel 工作表中。

MLGetVar：将 MATLAB 矩阵内容写到 Excel VBA 变量中。

MLPutMatrix：用 Excel 工作表中数据创建或覆盖 MATLAB 矩阵。

MLPutVar：用 Excel VBA 变量数据创建或覆盖 MATLAB 矩阵。

MLGetVar 和 MLPutVar 只能在宏中调用，其他函数可通过宏或工作表单元公式调用。

3．Excellink 实例

下面通过工作表对一组数据进行曲线拟合，说明 Excel link 的使用方法。

创建 Excel_example.xls 文件，如图 11-4 所示，DATA 下的第一列、第二列分别为数值 X 和 Y。

图 11-4 创建 Excel_example.xls 文件

选择单元 E4，按 F2 键，回车执行 Excel link 函数 MLPutMatrix("x",A4:A15)，它将 A4:A15 发送到 MATLAB 并给变量数组 *x* 赋值。对 E5 进行类似操作，将框中第二列数据赋值给 *y*。

对 E9、E10 执行类似操作，依次执行下列语句：

```
MLEvalString("p=polyfit(x',y',3)")
MLEvalString("newfit = polyval(p,x)")
```

这两条语句是 MATLAB 中执行双引号中命令，对数据点用 $y=ax^3+bx^2+cx+d$ 进行曲线拟合。

在 E11 中执行命令：

```
MLEvalString("a=p(1);b=p(2);c=p(3);d=p(4)")
```

该语句是 MATLAB 中将拟合系数分别给 *a*、*b*、*c*、*d* 赋值。

在 E14、E15、E16、E17 中分别执行命令：

```
MLGetMatrix("a","E21")
MLGetMatrix("b","F21")
MLGetMatrix("c","G21")
MLGetMatrix("d","H21")
```

这 4 条语句分别是 MATLAB 中将 *a*、*b*、*c*、*d* 传给 Excel 中单元格 E21、F21、G21、H21。在 E18 中执行命令：

```
MLEvalString("plot(x,newfit,'-' ,x,y,'o')")
```

该语句利用 MATLAB 绘图函数显示原始数据和拟合曲线，结果如图 11-5 所示。

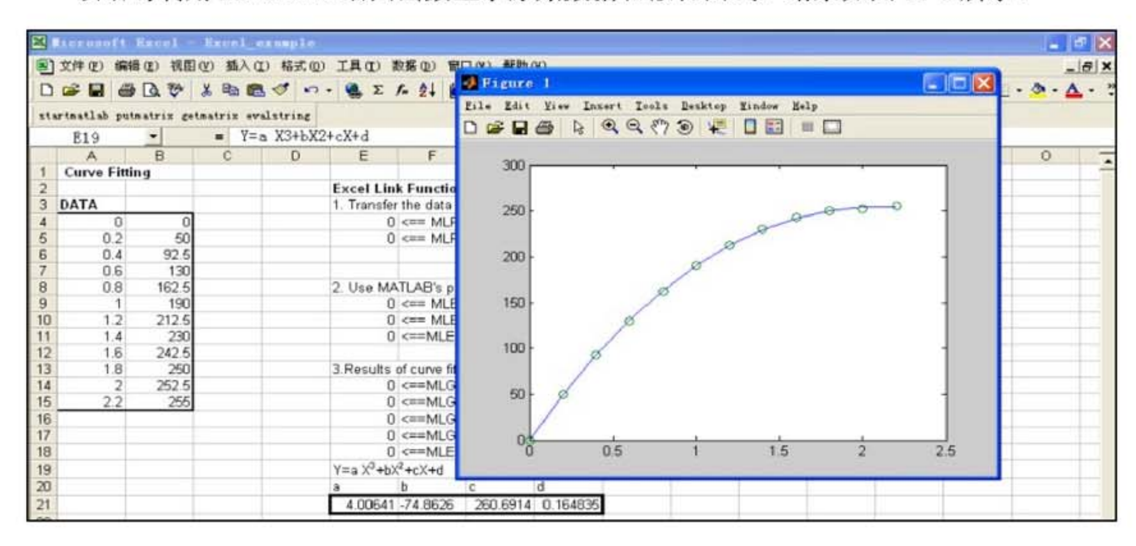


图 11-5　Excel Link 的运算结果

11.3　MATLAB 语言与 VB 接口

MATLAB 与 VB（Visual Basic）有多种接口方式，如 DDE、OLE、Active X 控件、COM 组件技术等，其中 COM 组件技术比较流行。下面介绍 MATLAB 7.0 的 COM 生成器，通过创建 COM 组件实现 MATLAB 与 VB 的接口。

11.3.1　COM 生成器

用 MATLAB COM 生成器创建 COM 组件，包括 4 个步骤，即创建工程、管理 m 文件和 mex 文件、生成组件、打包和分发组件。

1．创建工程

在 MATLAB 命令行中输入命令"comtool"，按 Enter 键后弹出【MATLAB Builder】窗口，如图 11-6 所示。单击菜单【File】的【New Project】选项，弹出【New Project Settings】（工程设置）窗口，如图 11-7 所示。

在【Component name】文本框中输入组件（DLL 文件）名称。输入组件名按 Enter 键后，生成器自动在【Class name】文本框中增加与组件名相同名称的类（class），可根据需要在【Class name】文本框修改类名。尽管组件名可以和类名一样，但组件名不能与 m 文件名或 mex 文件名相同。

把类添加到组件中，可在【Class name】文本框中输入类名，单击【Add】按钮，添加的类显示在【Classes】列表框中。

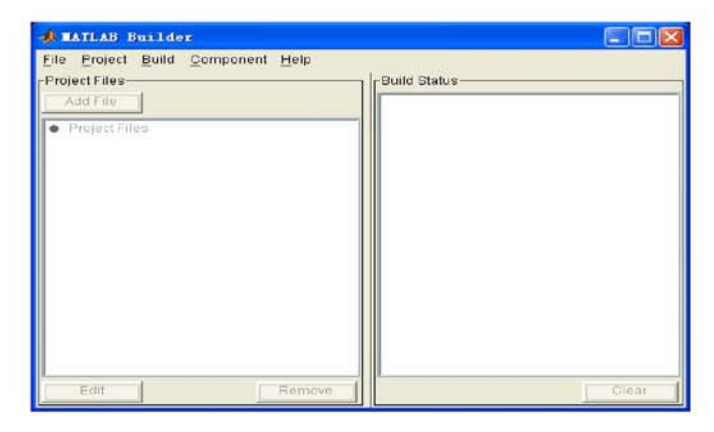


图 11-6 【MATLAB Builder】窗口

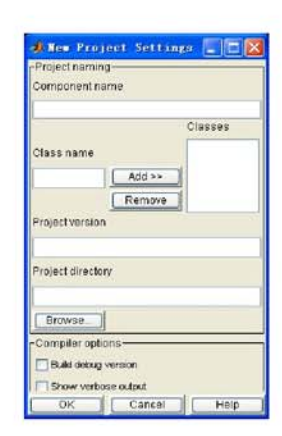


图 11-7 工程设置窗口

【Project version】文本框用于输入版本号（默认版本号为 1.0）。

【Project directory】文本框用于设置编译和打包模型时工程和相关文件存放位置，工程目录由当前目录和组件名自动组合生成。

设置完成后，单击【OK】按钮，它们成为用户工作空间的一部分，并与添加到工程中的 m 文件和 mex 文件一起保存，同时，工程目录中新增工程文件<component_name>.cbl。

2．管理 m 文件和 mex 文件

工程创建后，即可使用【MATLAB Builder】窗口中的【Project】、【Build】、【Component】菜单，如图 11-8 所示。单击【Add File】按钮或从【Project】菜单中选择【Add File】选项，可向工程中添加 m 文件或 mex 文件；单击【Remove】按钮或从【Project】菜单中选择【Remove】选项，可删除所选 m 文件或 mex 文件；选择 m 文件并单击【Edit】按钮或直接双击 m 文件名，可打开该 m 文件进行编辑和调试。

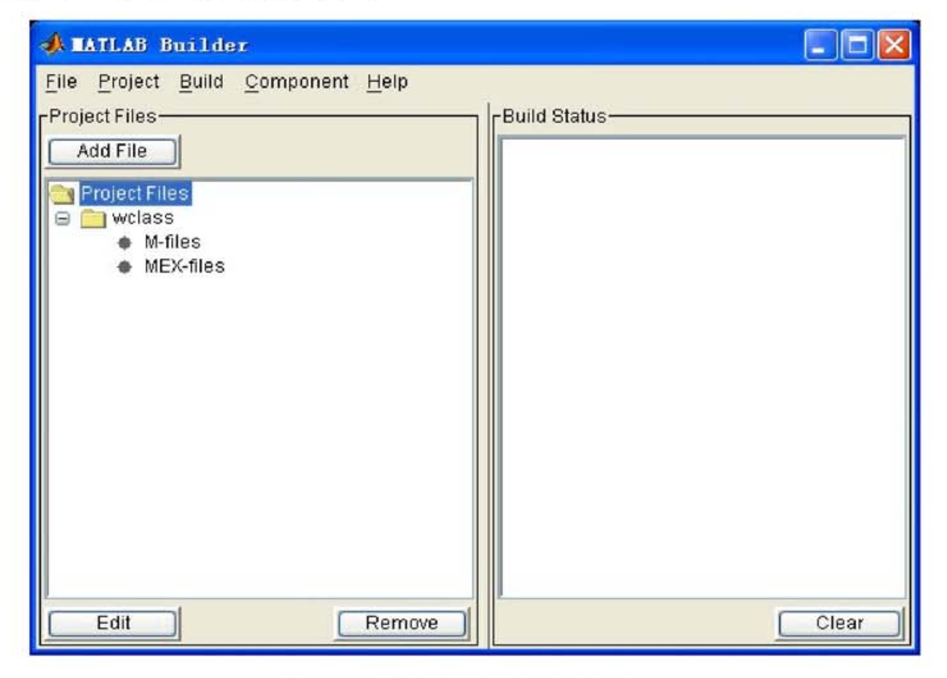


图 11-8 选项被激活的生成器主窗口

3．生成组件

工程设置完毕并添加有关 m 函数和 mex 函数后，选择菜单【Build】中的【COM Object】调用 MATLAB 编译器，可把中间源文件写入到<project_dir>\src 目录中，同时将必要的输出文件写到<project_dir>\distrib 目录中。

【Build Status】面板显示生成过程与遇到的问题，要清除【Build Status】面板，可单击【Build】菜单中的【Clear Status】。

生成过程保存在日志文件<project_dir>\build.log 中，可在【Build】菜单中选择【Open Build Log】打开该文件。

4．打包和分发组件

一旦编译成功并通过测试，就可以把有关文件打包分发给目标机器。

从【Component】菜单中选择【Package Component】，创建包含以下文件的自解压可执行程序（程序名为<componentname>.exe）。

_install.bat：自解压可执行程序运行的脚本。

<componentname_projectversion>.dll：编译后的组件。

mglinstaller.exe： MATLAB 数学库和图形库安装器。

mwcomutil.dll： COM 生成器工具库。

mwregsvr.exe：注册 DLL 的可执行程序。

运行安装器，须进行以下步骤：

- 通过 mglinstaller 安装 MATLAB C/C++数学和图形库，添加 mglinstaller 创建的目录<application>\bin\win32（<application>表示配置应用的根目录）。
- 通过 mwregsvr 注册 mwcomutil.dll 和<componentname>_<projectversion>.dll。

11.3.2　组件生成和应用实例

本例通过创建魔方矩阵的 m 文件，用 COM 编译器生成 COM 组件，嵌入到 VB 工程生成可执行应用程序。

1．创建 m 文件

将当前目录设置为 MATLAB 安装目录下的 work 目录，在 work 目录下创建文件 mymagic.m，其代码如下：

```
function y = mymagic(x)
y = magic(x);
```

2．创建工程

在命令窗口键入 comtool 命令，启动【MATLAB Builder】窗口。选择菜单【File】中的【New Project】，则弹出【New Project Settings】对话框，在该对话框进行如下设置。

- 【Component name】文本框中输入组件名“magicdemo”。
- 选择【Classes】列表框中自动生成的类 magicdemoclass，单击【Remove】删除。
- 输入类名“magic”，并单击【Add】按钮。
- 版本号默认为 1.0，不做修改。
- 【Project directory】文本框中包含 COM 生成器启动时的默认目录和组件名 magicdemo（可根据需要改变目录，目录不存在、系统将提醒创建）。设置完毕如图 11-9 所示。

- 单击【OK】按钮，创建 magicdemo 工程。

3．生成组件

- 在【MATLAB Builder】窗口选择【Magic】文件夹，单击【Add File】按钮。
- 选择文件 mymagic.m，单击【打开】按钮，将 m 文件加入工程，如图 11-10 所示。

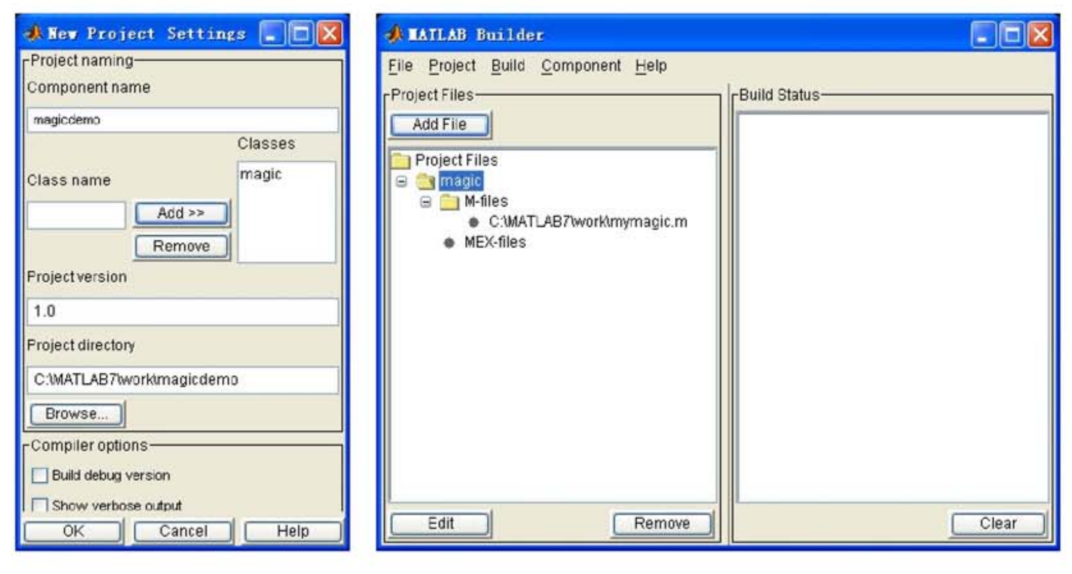


图 11-9　工程设置对话框　　　　图 11-10　工程中添加 M 文件

- 在菜单【Builde】中选择【COM Object】，创建组件（创建的组件保存在类目录的 distrib 子目录下），完成后的 COM 生成器如图 11-11 所示。

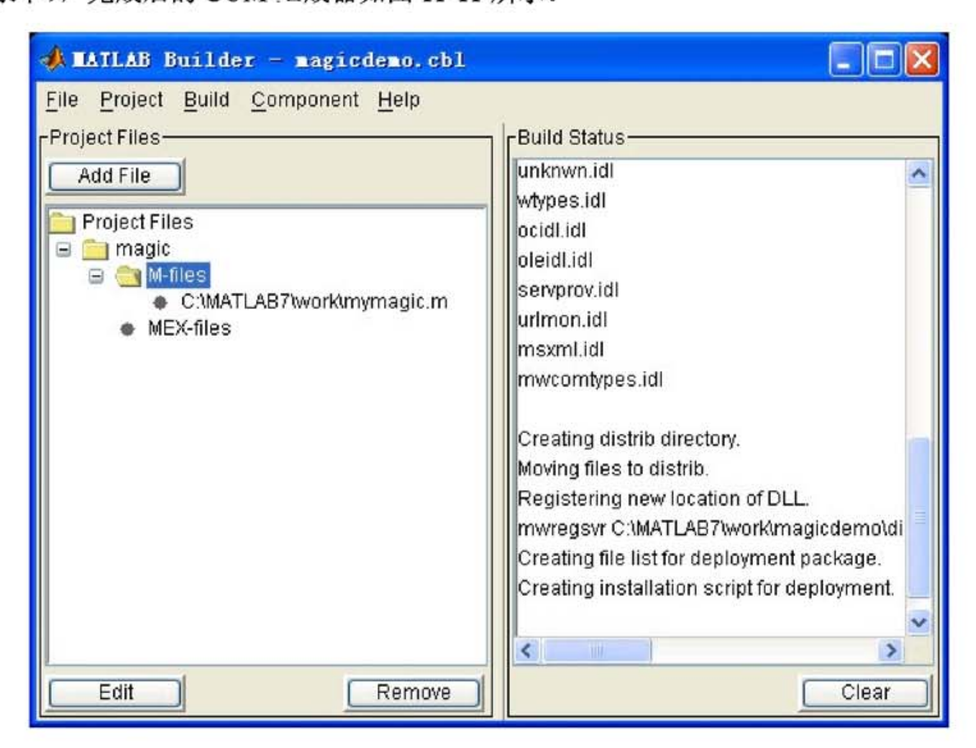


图 11-11　完成组件创建的 COM 生成器图形用户界面

4．创建 VB 工程并调用组件

创建 VB 工程并对 magicdemo 组件调用的过程如下。

➢ 新建“标准 EXE”工程，进入 VB 编程环境。

➢ 在【工程】菜单中单击【引用】选项，弹出【引用】对话框，在列表中选择【magicdemo 1.0 Type Library】，单击【确定】按钮。

➢ 返回到 VB 主菜单，在【工程】 菜单中单击【部件】选项，弹出【部件】对话框。

➢ 选择【Microsoft Windows Common Controls 6.0】，调用相关控件，单击【应用】按钮后关闭对话框。

11.4　VC 调用 MATLAB 引擎

MATLAB 引擎通过一组函数在应用程序中调用 MATLAB 程序环境进行计算和图形绘制，运行计算时将涉及 MATLAB 数组的调用与使用方法。本节在讲述引擎库函数和数组用法的基础上，通过实例说明 VC 调用 MATLAB 引擎的过程。

11.4.1 引擎库函数

MATLAB 引擎库函数的命名规则是名称前增加字符“eng”，使用功能包括以下几方面。

1．启动和关闭引擎

（1）engOpen 函数

函数声明为“Engine *engOpen(const char *startcmd);”，startcmd 为启动 MATLAB 进程的字符串（Windows 环境下为 NULL）。函数 engOpen(startcmd)返回指向引擎句柄的指针，作用是启动 MATLAB 进程、建立连接、返回唯一的引擎标别符（启动引擎失败返回 NULL）。

（2）engClose 函数

函数声明为“int engClose(Engine *ep);”，ep 为引擎指针。函数 engClose 的作用是关闭 MATLAB 引擎，返回 0 为关闭成功，返回 1 为关闭失败。

2．从 MATLAB 发送和接受数据

（1）engGetVariable 函数

函数声明为“mxArray *engGetVariable(Engine *ep, const char *name);”，ep 为事先打开的 MATLAB 引擎指针，name 为从 MATLAB 得到的变量。函数调用成功返回 mxArray 类型的指针，调用失败则返回 NULL。

（2）engPutVariable 函数

函数声明为“int engPutVariable(Engine *ep, const char *name, const mxArrray *mp);”，ep 为事先打开的 MATLAB 引擎指针，name 为写入引擎工作空间的变量名称，mp 为数组指针。函数 engPutVariable 把变量 mp 写入 ep 并指定变量名称 name，返回 0 表示调用成功、返回 1 则调用失败。

3．向 MATLAB 发送命令

（1）engEvalString 函数

函数声明为“int engEvalString(Engine *ep, const char *string);”，ep 为事先启动的引擎指针，string 为执行的字符串。函数执行成功返回 0，若 ep 对应引擎已关闭则返回 1。

（2）engOutputBuffer 函数

函数声明为“int engOutputBuffer(Engine *ep, char *p, int n);”，ep 为事先启动的 MATLAB 引擎指针，p 为字符串缓冲区指针，n 为 p 的大小。engOutputBuffer 为函数 engEvalString 返回值定义字符串缓冲区，停止缓冲可使用命令“engOutputBuffer(ep, NULL,0);”。

4．其他

（1）engGetVisible 函数

函数声明为“int engGetVisible (Engine *ep, bool *value);”，ep 为事先打开的引擎指针，value 为指向 engGetVisible 返回值的指针。该函数确定当前 MATLAB 窗口是否可见。

（2）engSetVisible 函数

函数声明为“int engSetVisible(Engine *ep, bool value);”，ep 为事先打开的 MATLAB 引擎指针，value 为 MATLAB 设定的可见属性。该函数显示或隐藏 MATLAB 窗口，value 属性值为 1 表示 MATLAB 窗口可见，为 0 表示 MATLAB 窗口不可见。

11.4.2 MATLAB 数组的用法

用 C 语言编制程序文件和使用 MATLAB 引擎库函数调用 MATLAB 实现计算与绘图功能时，离不开对 MATLAB 数组的操作。MATLAB 的 engine 程序、mex 程序和 MATLAB C 数学库中最常用的数据结构是 mxArray，它包括数组类型、维数、数据等。MATLAB 外部接口函数库中提供了一系列 mex 函数，实现 C 语言中创建、操作、清除 mxArray 结构体对象，下面分别介绍。

1．数组创建和清除

MATLAB 的变量数据类型有多种，每种数据类型都由对应函数创建，一般格式是：mxCreatxxx。创建二维 double 型数组，可采用下面代码：

```
#include “matrix.h”
mxArray *mxCreateDoubleMatrix(int m, int n, mxComplexity ComplexFlag);
```

其中，*m* 表示矩阵数组行数，*n* 表示矩阵数组列数，ComplexFlag 为常数（取 mxREAL 和 mxCOMPLEX 时矩阵中数据分别为实数和复数）。第一句为包含头文件 matrix.h（写 engine 程序时一般还要包含头文件 engine.h），第二句为创建数组。函数调用成功返回一个指向数据结构 Array 的指针，否则返回 NULL。

数组使用完毕，可以使用函数 mxDestroyArray 把数组从内存中删除，该函数声明为“void mxDestroyArray(mxArray *array_ptr);”，array_ptr 为要删除的数组指针。

常用的数组创建函数还有如下几种：

```
mxArray *mxCreateCellArray(int ndim, const int *dims);   //创建多维单元数组
mxArray *mxCreateCellMatrix(int m, int n);                //创建二维单元数组
mxArray *mxCreateCharArray(int ndim, const int *dims);   //创建多维字符数组
mxArray *mxCreateString(const char *str);                //创建字符串
mxArray *mxCreateStructMatrix(int m, int n, int nfields, const char **field_names);
                                                          //创建结构体矩阵
```

2．数组类型判断

使用数组变量前需要判断其类型，判断数组变量类型是否为字符串的判断函数为“bool

mxIsChar(const mxArray *array_ptr);”，array_ptr 为被判断数组的指针。判断函数还有如下几种：

```
bool mxIsDouble(const mxArray *array_ptr);                    // 判断是否为 double 数组
bool mxIsEmpty(const mxArray *array_ptr);                     // 判断是否为空数组
bool mxIsClass(const mxArray *array_ptr, const char *name);   // 判断是否为某种类
bool mxIsStruct(const mxArray *array_ptr);                    // 判断是否为某种结构
```

3．数组维数管理

要得到数组每一维元素的个数，可采用 mxGetM 和 mxGetN 函数实现。mxGetM 获取数组第一维的元素个数（对二维数组就是矩阵的行数），mxGetN 获得其他维的元素个数的乘积（对二维数组就是矩阵的列数），函数声明分别为（array_ptr 为指向矩阵的指针）：

```
int mxGetM(const mxArray *array_ptr);
int mxGetN(const mxArray *array_ptr);
```

要得到数组某特定维的元素个数，可使用 mxGetDimensions 函数实现，函数声明为“const int *mxGetDimensions(const mxArray *array_ptr);”，各维元素个数保存在返回的整数数组中。

常用的维数管理函数还有如下几种：

```
int mxGetNumberOfDimensions(const mxArray *array_ptr);     // 获得数组总维数
void mxSetM(mxArray *array_ptr, int m);                     // 设定矩阵行数
void mxSetN(mxArray *array_ptr, int n);                     // 设定矩阵列数
```

4．数组数据管理

MATLAB 数组管理函数对数组数据进行读写的一般格式为 mxGetxxx 和 mxSetxxx。常用的数据管理函数有如下几种：

```
mxClassID mxGetClassID(const mxArray *array_ptr);
                                    //获取结构体 mxArray 的类型（枚举类型常量）
const char *mxGetClassName(const mxArray *array_ptr);
                                    //获取结构体 mxArray 的类型（字符串）
int mxGetString(const mxArray *array_ptr, char *buf, int buflen);   // 获取字符串
void *mxGetData(const mxArray *array_ptr);                          // 获取数组数据
mxArray *mxGetCell(const mxArray *array_ptr, int index);            //获取单元内容
mxArray *mxGetField(const mxArray *array_ptr, int index, const char *field_name);
                                    //通过指定域获取域的内容
void mxSetData(mxArray *array_ptr, void *data_ptr);                 // 设置数组数据
void mxSetCell(mxArray *array_ptr, int index, mxArray *value);      // 设置单元内容
```

MATLAB 的数组操作函数还有很多，读者可参阅 MATLAB 帮助文档。

11.4.3　VC 调用 MATLAB 引擎使用实例

1．设置 VC 的头文件搜索路径和库文件搜索路径

选择【Tool】菜单中的【Options】，选择【Directories】（路径）属性页，分别添加头文件和库文件搜索路径%matlabroot%\extern\include、%matlabroot%\extern\lib\ win32\ microsoft\ msvc60，如图 11-12 和图 11-13 所示。



图 11-12　头文件搜索路径设置　　　　图 11-13　库文件搜索路径设置

2．建立工程

在指定目录下用 MFC AppWizard(exe)方式建立名为“engwm”的工程（如图 11-14 所示），选择【确定】后，选择单文档、支持文档/视结构（见图 11-15），其他各对话框选择默认值，最后单击【完成】，得到新工程 engwm。

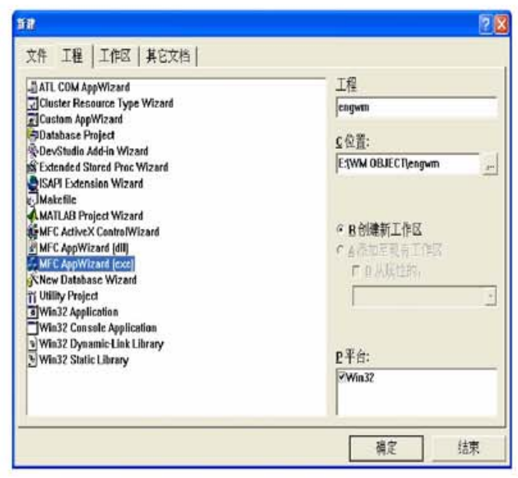


图 11-14　建立工程 engwm



图 11-15　选择应用程序类型

（1）添加 C++文件

编写数据插值程序 Data_interpolation.cpp 并加入工程，程序代码如下：

```
#include "stdafx.h"
#include "engine.h"
void interpolation()
{
    Engine *wm;
    if(!(wm=engOpen(NULL)))
    MessageBox(NULL,"无法打开 MATLAB 引擎","数据插值",MB_OK);
    engEvalString(wm,"rand('seed',0);
    x = rand(100,1)*4-2; y = rand(100,1)*4-2; ");
    engEvalString(wm,"z = 100*(y.^3-x.^2);");
```

```
        engEvalString(wm,"ti = 0:0.05:2; ");
        engEvalString(wm,"[XI,YI] = meshgrid(ti,ti);");
        engEvalString(wm,"ZI = griddata(x,y,z,XI,YI);");
        engEvalString(wm,"surf(XI,YI,ZI);
        title('数据插值三维曲面图');xlabel('x');ylabel('y');zlabel('z');grid on");
        MessageBox(NULL,"插值计算完毕","数据插值",MB_OK);
    }
```

（2）添加菜单项

VC 工作窗口中选择 Resource 栏并打开菜单编辑器，如图 11-16 所示。

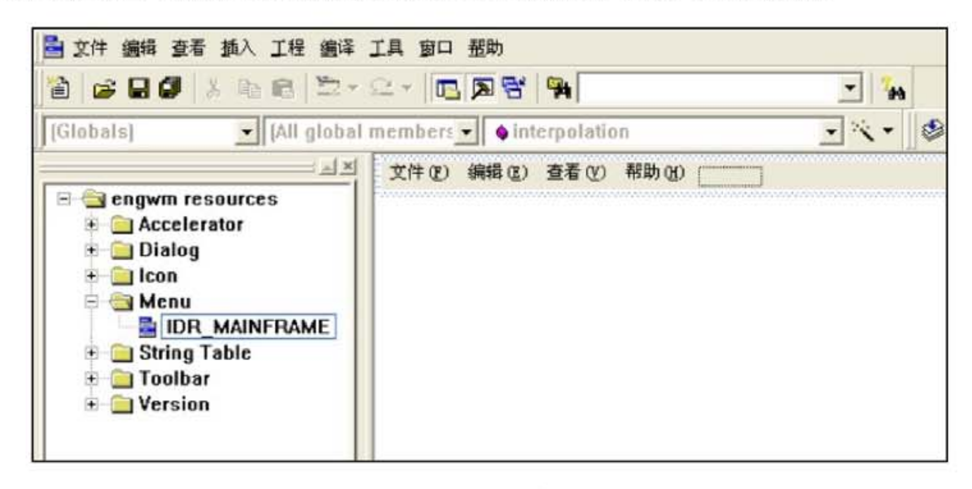


图 11-16　菜单编辑器

双击图 11-16 的虚线框，在弹出对话框【标题】文本框输入“数据插值”（见图 11-17），同理增设子菜单项为“执行”（ID 为 ID_DATA_ INTERPOLATION，见图 11-18）。

（3）编写消息处理函数

打开 ClassWizard，为菜单增加消息处理函数。将【执行】菜单的消息处理交给类 CMainFrame 完成，Object IDS 列表中选择 ID_DATA_INTERPOLATION，在【Class Name】列表框中选择“CMainFrame”，【Message】列表框中选择“COMMAND”，单击【Add Function】按钮，在弹出的函数名确认对话框中单击【OK】，如图 11-19 所示，再单击【Edit Code】按钮，为 CMainFrame 添加消息处理函数 OnDataInterpolation，其代码如下：

```
......
void CmainFrame::OnCurvePlot()
{
//TODO:Add your command handler code here
interpolation();
}
......
```

在 MainFrm.cpp 文件开始加上函数声明：

```
void interpolation();
```

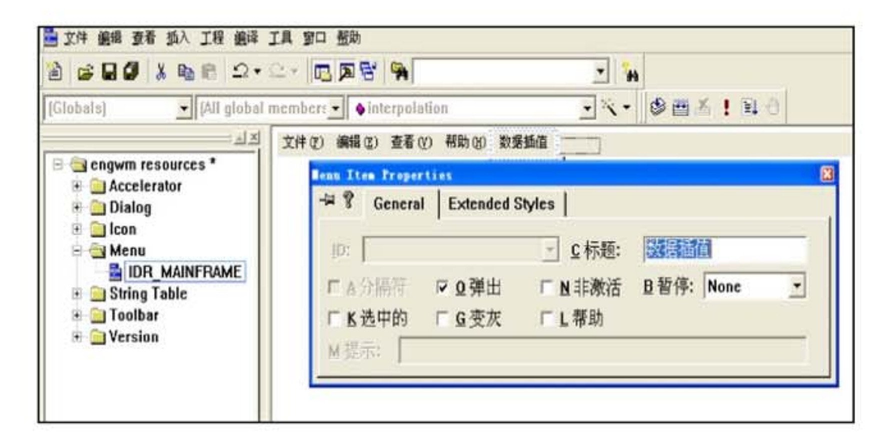


图 11-17 编辑“数据插值”菜单

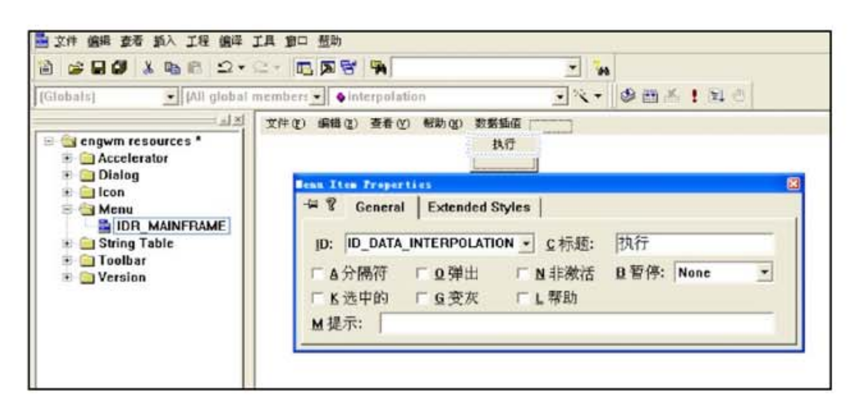


图 11-18 编辑子菜单

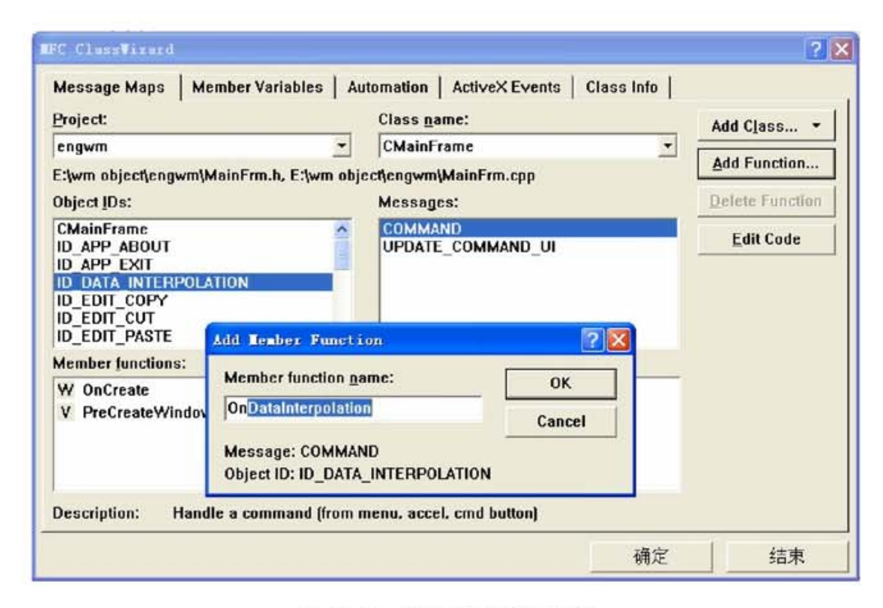


图 11-19 添加消息处理函数

（4）设置工程选项

选择【Project】菜单中的【Settings】，打开工程设置对话框，在【Link】选项卡的【对象

/库模块】文本框内加入库文件名 libmx.lib libmat.lib libeng.lib，如图 11-20 所示。

图 11-20 添加库文件

设置完毕并编译连接后，在工程目录 engwm 的文件夹 debug 里生成可执行文件 engwm.exe，双击此文件弹出插值程序窗口，再双击【数据插值】菜单中的【执行】菜单项，可以得到插值计算结果，如图 11-21 所示。

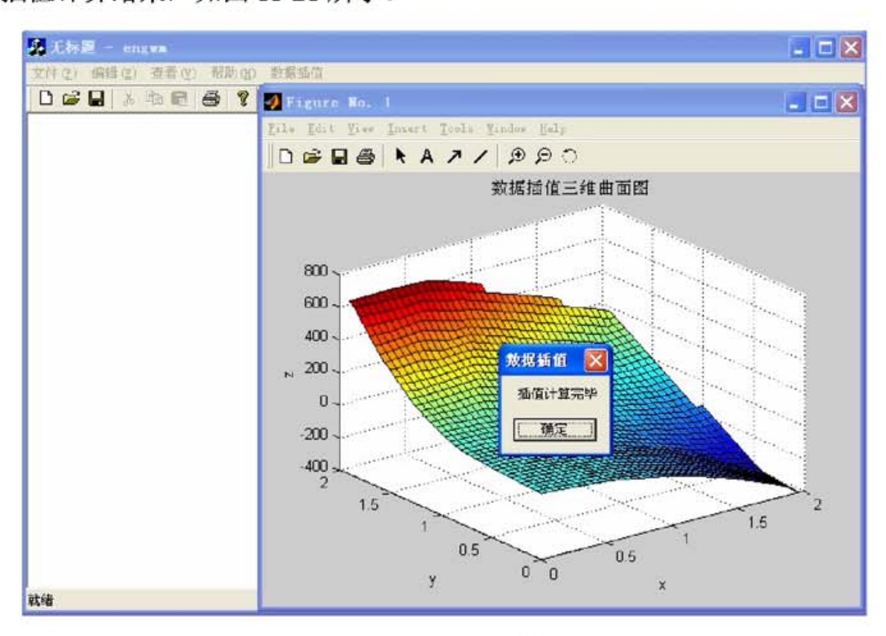


图 11-21 插值计算结果

11.5 VC 编译 MATLAB 的 mex 文件

mex 文件是 MATLAB 调用其他程序设计语言程序或算法的接口。在 Windows 环境中，mex 文件是扩展文件名为 DLL 的动态链接库，可以在 m 程序中直接调用，用户可以通过

MATLAB 提供的命令“mex”将 C 程序编译成 mex 文件，实现 MATLAB 环境下直接调用或链接这些程序。mex 文件必须在 C 语言编写后用“mex”命令编译生成，编写时按照 MATLAB 规定的格式和步骤进行，采用 mex 函数库中的函数与 MATLAB 进行交互，从 MATLAB 中获取数据并返回信息。

本节在讲述 mex 文件系统设置、mex 函数和 mex 文件建立方法的基础上，通过实例说明 VC 编译 MATLAB 中 mex 文件的方法。

11.5.1 mex 文件系统设置

编译和生成 mex 文件，需要安装 MATLAB 应用程序接口组件及 C 语言编译器。在 Windows 环境下编译 mex 文件，编译器必须支持 32 位的 Windows 动态链接库。第一次使用 mex 命令时必须通过命令窗口执行命令“mex –setup”进行适当配置，配置时第一个问题可选“n”，编译器可选用“Microsoft Visual C/C++ version 6.0”。

11.5.2 mex 函数和 mex 文件

mex-函数是 MATLAB 外部程序的接口函数，以 mex 为前缀，用于实现 mex 文件与 MATLAB 的交互，mexFunction 函数是 mex 文件的核心。

MexFunction 的函数声明如下：

```
void mexFunction(int nlhs,mxArray *plhs[ ], int nrhs,const mxArray *prhs[ ])
```

其中，nlhs 为输出变量的个数；plhs 为指向输出变量指针的数组；nrhs 为输入变量的个数；prhs 为指向输入变量指针的数组。mexFunction 函数中的参数声明是只读的，不能被修改，MATLAB 调用 mex 程序时自动生成参数 nlhs、plhs、nrhs 和 prhs。

MATALB 中相应函数的调用格式为“[a,b,…]=fun(c,d,…)”，a，b，… 为函数的输出变量，c，d，… 为函数的输入变量，此时 nlhs 为 a，b，… 变量的个数，nrhs 为 c，d，… 变量的个数。例如，调用名为 engwm 的文件，可在 MATLAB 的命令行输入“[a,b,c]=engwm(x,y)”，执行后 nrhs=2、nlhs=3、并创建指针数组 prhs[0]-> x、prhs[1]->y，这时输出变量还没有创建，指针数组中 plhs[0]、plhs[1]、plhs[2]均指向 NULL。

函数 mexErrMsgTxt 用于显示错误信息，显示完毕后立即终止当前运行的 mex 程序（这与其他显示函数不同）。其他 mex 函数可参阅 MATLAB 的帮助文档。

下面通过编写显示字符串的程序，说明 mex 文件的建立和使用方法。

（1）编写 C 语言程序代码

VC 中新建文件 good.cpp 并将其保存在指定目录中（如 E:\MEX），如图 11-22 所示。good.cpp 的代码如下：

```
#include “mex.h”

void mexFunction(int nlhs,mxArray *plhs[ ],int nrhs, const mxArray *prhs[ ])
{
mexPrintf(“Good Morning!\n”);
}
```

这里，必须包含头文件 mex.h（该文件包含定义矩阵的头文件 matrix.h 和 mex 函数的声

明），mexPrintf 函数（只能用于 mex 程序）与 printf 函数的用法基本相同。

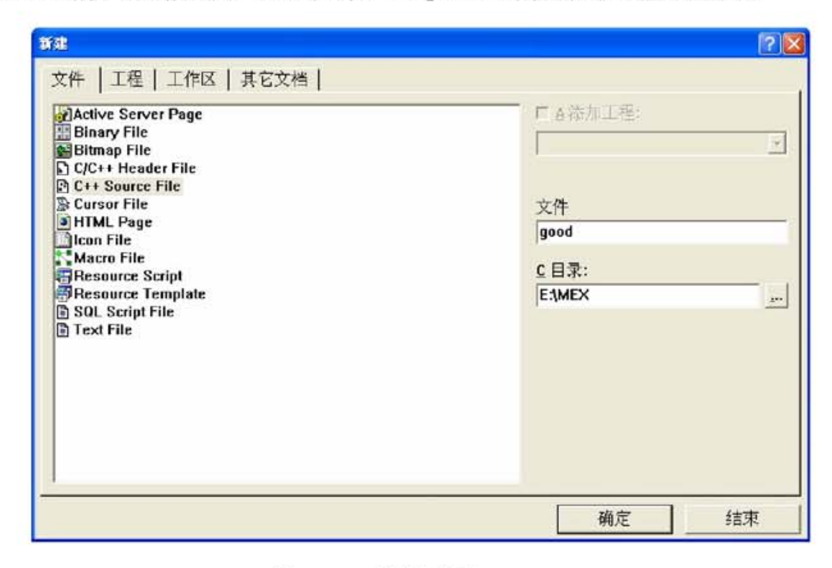


图 11-22　创建文件 good.cpp

（2）编译 good.cpp

打开 MATLAB 平台并将工作目录改为 good.cpp 所在目录 E:\MEX，输入命令“mex good.cpp”，则在目录 E:\MEX 下生成同名 DLL 程序 good.dll。

（3）执行 good.dll 程序

输入命令“good”将显示字符串“Good Morning”，如图 11-23 所示。

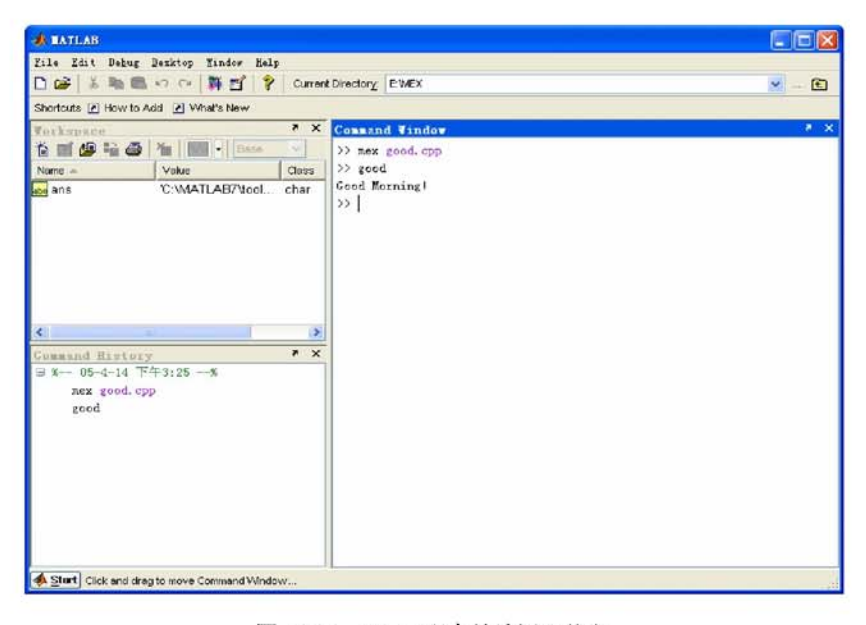


图 11-23　MEX 程序的编译和执行

上述源文件 good.cpp 为 C++文件，也可用 C 文件 good.c 作为源文件（两文件代码相同）。用 mex 编译生成 mex 文件时，命令改为“mex good.c”，good.c 所在目录中生成 good.dll，而 good.dll 的调用方法不变。

11.5.3 VC 编译 mex 文件使用实例

前面通过 MATALB 命令“mex”编译了 mex 文件，如果 mex 程序比较复杂（例如需要复杂界面窗口），应考虑使用 VC 的 Project Wizard、Class Wizard 编译，下面用一个实例介绍。

1．设置 VC 的头文件搜索路径和库文件搜索路径

设置方法与 VC 调用 MATLAB 引擎时的设置相同，这里不再赘述。

2．建立工程

用 MFC AppWizard(dll)方式在指定目录下建立名为 Mexwm 的工程并选择静态链接。工程 Mexwm 包含若干头文件和 Mexwm.cpp、Mexwm.def、Mexwm.rc、StdAfx.cpp 文件。

文件 Mexwm.def 中加入 mexFunction 函数，修改后的代码如下：

```
; Mexwm.def : Declares the module parameters for the DLL.
LIBRARY         "Mexwm"
DESCRIPTION   'Mexwm Windows Dynamic Link Library'
EXPORTS
      ; Explicit exports can go here
mexFunction
```

新建文件 circumference.cpp 并加入工程（文件功能是求圆周长），代码如下：

```
#include "stdafx.h"
#include "mex.h"
#include <math.h>
void circumference(double y[ ],double x[ ])
{
      y[0]=x[0]*2*3.14;
      return;
}

void mexFunction(int nlhs,mxArray *plhs[],int nrhs,const mxArray *prhs[])
{
      double *x;  double *y; CString str;
//检查变量个数
        if(nrhs!=1)
          {
            mexErrMsgTxt("输入变量只能为 1 个！ ");
          }
         else if(nlhs!=1)
          {
            mexErrMsgTxt("输出变量只能为 1 个！ ");
          }
        plhs[0]=mxCreateDoubleMatrix(1,1,mxREAL); //创建输出变量矩阵
        y=mxGetPr(plhs[0]);   //赋输出变量的数据指针给 y
        x=mxGetPr(prhs[0]);   //赋输入变量的数据指针给 x
//调用计算程序
        circumference(y,x);
        str.Format("圆周为%3.2f",*y);
```

```
        AfxMessageBox(str);
    }
```

3．设置工程选项

选择菜单【Project】中的【settings】打开工程设置对话框，在 Link 选项卡的【对象/库模块】文本框输入库文件名 libmx.lib libmex.lib libmat.lib（见图 11-24），在【C/C++】选项卡的【Preprocessor definitions field】文本框中输入 MATLAB_MEX_FILE（见图 11-25）。设置完毕并编译连接之后，工程所在目录下 debug 文件夹中生成动态连接库 Mexwm.dll。

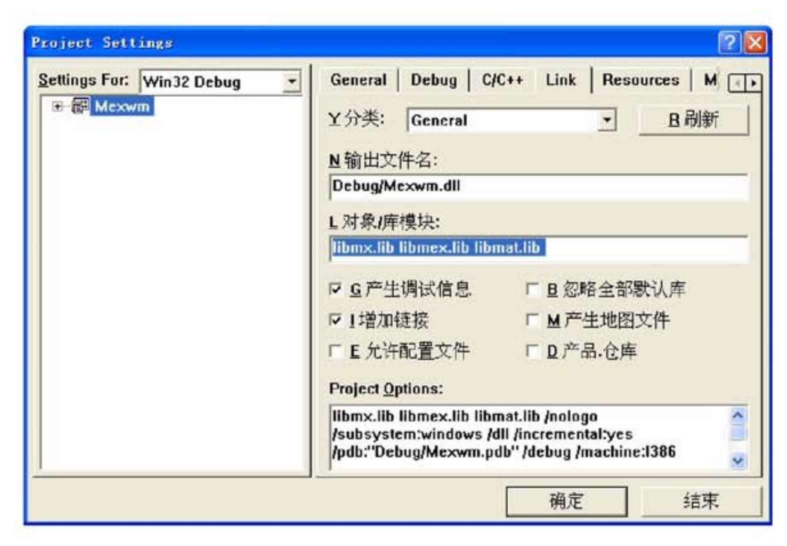


图 11-24　添加库文件

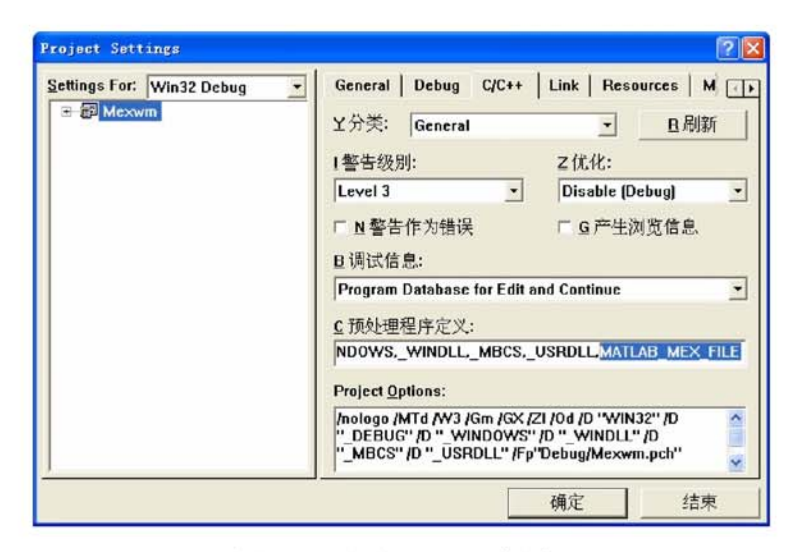


图 11-25　添加预处理程序定义

4．运行程序

打开 MATLAB 平台并将当前目录设为 Mexwm.dll 所在目录 E:\wm object\ Mexwm\ Debug，输入命令 “x=6;y=Mexwm(x);” 得到圆周结果，如图 11-26 所示。

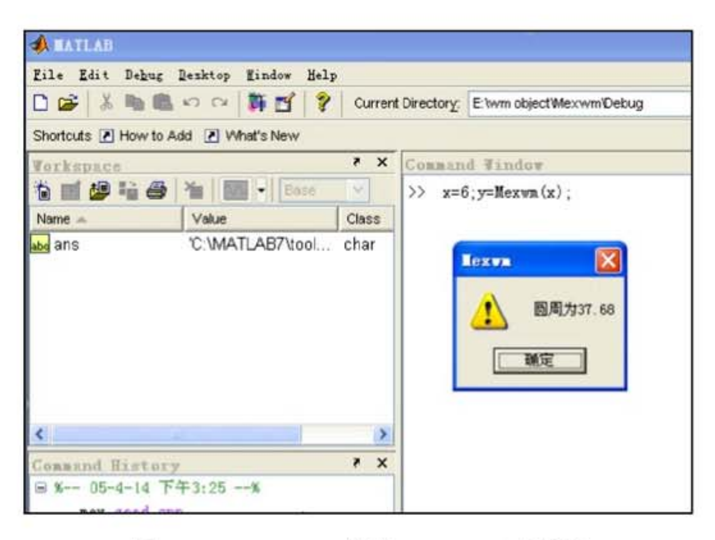


图 11-26 MATLAB 执行 Mexwm.dll 的结果

习题 11

1．列举 MATLAB 编译器 4.0 与以往编译器的不同。

2．试通过 MATLAB 自带帮助文档，归纳 mcc 命令的用法。

3．利用 MATLAB 编译器生成的独立应用程序分发到目标机器有哪些步骤？

4．独立应用程序如何在目标机器上安装和运行？

5．试述 EXCEL Link 的安装步骤。

6．列举 EXCEL Link 的数据处理函数及其功能。

7．用 COM 生成器生成和创建 COM 组件，包括哪四个步骤？

8．组件打包时，应将那些文件打包创建自解压可执行程序？

9．MATLAB 引擎库函数常用的有哪些？请说明相应的功能。

10．VC 调用 MATLAB 引擎时，需进行哪些设置?与 VC 中编译 mex 文件时的设置有什么不同?

11．已知 A 市有甲、乙、丙、丁四县。2004 年各县外贸出口额分别为 4 千万元，6 千万元，5.5 千万元 9 千万元，请用 VC 调用 MATLAB 引擎绘制饼状图显示各县对 A 市外贸出口贡献率。

12．用 VC++编译生成 mex 文件，实现输入长方体的长、宽、高的功能，并求出其体积。

13．利用 EXCEL Link 调用 MATLAB，对矩阵 A=[1 3 5；4 6 8；8 3 9]进行 LU 分解，并将结果显示在 EXCEL 列表中。

附录 A　MATLAB 7.0 基本命令函数一览

MATLAB 系统提供了大量的命令函数，包括内部命令和 m 文件形式的函数，其安装目录的不同子目录下安装了这些 m 文件并在 content.m 文件中对其作了简单介绍，为方便读者全面使用命令函数，下面以一览表的形式列出了其中比较常用的一些命令，读者可使用命令“help fun”(fun 为.m 文件名)获得各个文件的简要说明。

表 A-1　MATLAB 基本命令函数一览表

<table>
<tr><th>目 录 名</th><th>命 令 类 型</th><th>命令函数一览</th></tr>
<tr><td>general</td><td>通用命令</td><td>addpath，demo，doc，docopt，genpath，help，helpbrowser，helpdesk，lasterr，lastwarn，license，lookfor，partialpath，path，pathtool，profile，rehash，rmpath，support，type，what，which，clear，disp，length，load，memory，mlock，munlock，openvar，pack，pack，save，saveas，size，who，whos，workspace，!，cd，delete，diary，dir，getenv，unix，cedit，clc，echo，format，home，more，exit，finish，matlab，matlabbrc，quit，startup，hostid，info，subscribe，ver，version，web，whatsnew</td></tr>
<tr><td>ops</td><td>操作符和特殊字符</td><td>+，-，!，%，&，()，*，,，.，.`，..*，..，./，.\，.^，/，:，,，;，[]，\，^，{}，|，，''，<>，=，==，...，kron，xor</td></tr>
<tr><td>logfun</td><td>逻辑函数</td><td>all，any，exist，find，is，isa，isempt，isglobal，iskeyword，isreal，issparse，isstr，isvarname，logical，mislocked</td></tr>
<tr><td>lang</td><td>语言结构和调试</td><td>builtin，eval，evalc，evalin，feval，function，global，nargchk，persistent，script，break，case，catch，continue，else，elseif，end，error，for，if，otherwise，return，switch，try，warning，while，input，keyboard，menu，pause，class，double，inferiorto，inline，int16，int32，int8，isa，loadobj，saveobj，single，superiorto，uint16，uint32，uint8，dbclear，dbcont，dbdown，dbmex，dbquit，dbstack，dbstatus，dbstep，dbstop，dbtype，dbup，func2str，function，str2func</td></tr>
<tr><td>elmat</td><td>基本矩阵和矩阵操作</td><td>:，eye，linspace，logspace，meshgrid，ones，rand，randn，zeros，ans，blkdiag，computer，eps，flops，i，inf，inputname，isieee，j，nan，nargin，nargout，nargoutchk，pi，realmax，realmin，varargin，varargout，version，why，calendar，clock，cputime，date，datenum，datestr，datevec，eomday，etime，now，tic，toc，weekday，:，cat，diag，fliplr，flipud，repmat，reshape，rot90，tril，triu，cross，dot，intersect，ismember，setdiff，setxor，uion，unique</td></tr>
<tr><td>specmat</td><td>特殊矩阵</td><td>compan，gallery，hadamard，hankel，hilb，invhilb，kron，magic，pascal，rosser，toeplitz，vander，wilkinson</td></tr>
<tr><td>elfun</td><td>基本数学函数</td><td>acos，acosh，acot，acoth，acsc，acsch，asec，asech，asin，asinh，atan，atan2，atanh，cos，cosh，cot，coth，csc，csch，sec，sech，sin，sinh，tan，tanh，exp，log，log，sqrt，abs，angle，complex，conj，image，real，ceil，fix，floor，gcd，lcm，mod，nchoosek，rem，round，sign，bvp4c，bvpget，bvpinit，bvpset，bvpval，colamd，convhulln，delaunay3，delaunayn，dsearchn，griddata3，griddatan，isqr，minres，pchip，pdepe，pdeval，quadl，symamd，symmlq，varonoin</td></tr>
</table>

续表

目 录 名	命 令 类 型	命令函数一览
specfun	特殊数学函数	airy，besseli，besselj，besselk，bessely，beta，betainc，betaln，ellipj，ellipke，erf，erfc，erfcx，erfinv，epxint，factorial，gamma，gammainc，gammaln，legendre，log2，pow2，rat，rats
corconv	坐标系统变换	cart2pol，cart2sph，pol2cart，sph2cart
matfun	数值线性代数	cibdeig，cond，det，norm，null，orth，rank，rcond，rref，rref，rrefmovie，subspace，trace，\，/，chol，inv，isqnonneg，lscov，lu，minres，nnls，pinv，qr，qrdelete，qrinsert，eymmlq，banlance，cdf2rdf，eig，hess，poly，polyeig，qsvd，qz，rsf2csf，schur，svd，expm，expm1，expm2，expm3，funm，logm，sqrtm，qrdelete，qrinsert
datafun	数据分析和傅里叶变换	cumprod，cumsum，cumtrapz，factor，inpolygon，max，mean，median，min，perms，polyarea，primes，prod，rectint，sort，sortrows，std，sum，sumprod，trapz，var，del2，diff，gradient，cross，dot，corrcoef，cov，subspace，conv，conv2，deconv，filter，filter2，abs，angle，cplxpair，fft，fft2，fftshift，ifft，ifft2，ifftn，ifftshift，nextpow2，unwrat
polyfun	多项式和内插函数	conv，deconv，poly，polyder，polyeig，polyfit，polyint，polyval，polyvalm，residue，residue，root，convhull，convhulln，delaunay，delaunay3，delaunayn，dsearch，dsearchn，griddata，driddata3，griddatan，interp1，interp2，interp3，interpft，interpn，meshgrid，ndgrid，pchip，ppval，spline，tsearch，tsearchn，voronoi，voronoin
funfun	泛函-非线性数值方法	ode23，bcpinit，bcpval，bvp4c，bvpget，bvpset，dblquad，fmin，fminbnd，fmins，fminsearch，fplot，fzero，ode23p，ode45，odeget，odeset，optimget，optimset，pdepe，pdeval，quad，quad1，quad8，vectorize
sparfun	稀疏矩阵	speye，sprand，sprandn，sprandsym，spdiags，find，full，sparse，spconvert，issparse，nnz，nonzeros，nzmax，spalloc，spfun，spones，gplot，spy，colamd，colmmd，colperm，dmperm，symamd，symmmd，symrcm，randperm，condest，normest，sprand，treelayout，etree，etreeplot，treeplot，bicg，bicgstab，cgs，chlupdate，cholinc，gmres，lsqr，luinc，pcg，qmr，qr，qrdelete，qrinsert，qrupdate，eigs，svds，spaugment，spparms，symbfact
sounds	音频处理	lin2mu，mu2lin，sound，soundsc，auwrite，auread，wavplay，wavread，wavrecord，wavwrite
strfun	字符串函数	abs，eval，real，strings，blanks，deblank，func2str，isstr，setstr，str2func，str2mat，deblank，findstr，isletter isspace，lower，strcat，strcmp，strcmp，strjust，strncmp，strncmpi，strrep，strtok，strvcat，symvar，textlabel，upper，char，int2str，mat2str，num2str，sprintf，sscanf，str2double，str2mat，str2num，bin2dec，dec2bin，dec2hex，hex2dec，hex2num
iofun	I/O 函数	fclose，fopen，fread，fwrite，feof，ferrer，frewind，fseek，ftell，fget1，fgets，fprintf，fscanf，sprintf，sscanf，csvread，csvwrite，dlmread，dlmwrite，hdf，imfinfo，imread，imwrite，strread，texread，wk1const，wk1read，wk1wrec，wk1write，serial，fgetl，fgets，fprintf，fread，fscanf，fwrite，readsaync，stopasync，get，set，clear，delete，disp，fcolse，fopen，instraction，instrfind，isvalid，length，load，record，save，seialbreak，size

续表

目　录　名	命　令　类　型	命令函数一览
bitfun	位函数	bitand，bitcmp，bitget，bitmax，bitor，bitset，bitshift，bitxor
Structfun	结构函数	fieldnames，getfield，rmfield，setfield，struct，struct2cell
objfun	对象函数	class，isa，methods，methodsview，subasgn，subsindex，subsref
inttojava	JAVA 界面	class，import，isa，isjava，javaArray，javaMethud，javaObject，methods，methodsview
cellarrfun	单元数组	cell，cell2struct，celldisp，cellfun，cellplot，cellstr，num2cell
mutiarrfun	多维数组	cat，flipdim，ind2sub，ipermute，ndims，ndqrid，permute，reshape，shiftdim，squeeze，sub2ind
plotdatvis	绘图与数据可视化	bar，barh，hist，histc，hold，loglog，pie，plot，polar，semilogx，semilogy，subplot，bar3，bar3h，comet3，cylinder，fill3，plot3，quiver3，slice，sphere，stem3，waterfall，clabel，datetick，grid，gtext，legend，plotyy，title，xlabel，ylabel，zlabel，contour，contourc，contourf，hidden，mesh，meshc，peaks，surf，surface，surfc，surfl，trimesh，trisurf ， coneplot ， contourslice ， curl ， divergence ， flow ，interpstreamspeed ， isocaps ， isocolors ， isonormals ， isosurface ，reducepatch，reducevolume，shrinkfaces，slice，smooth3，stream2，stream3 ， streamline ， streamparticles ， streamribbon ， streamslice ，streamtube，surt2patch，subolume，volumebounds，griddata，meshgrid，area，box，comet，compass，convhull，delaunay，dsearch，errorbar，ezcontour，zcontourf，ezmesh，ezmeshc，ezplot，ezplot3，ezpolar，ezsurf，ezsurfc，feather，fill，fplot，inpolygon，pareto，pcolor，pie3，plotmatrix，polyarea，quiver，ribbon，rose，scatter，scatter3，stairs，stem，tsearch，voronoi，camdolly，camlookat，camorbit，campan，campos，camproi，camroll，camtarget，camup，camva，camzoom，daspect，pbaspect，view，viewmtx，xlim，ylim，zlim，camlight，light，lightangle，material，alim，alpha，alphamap，brighten，caxis，colorbar，colordef，colormap，graymon，hsv2rgb，rgb2hsv，rgbplot，shading，spinmap，surfnorm，whitebg，autumn，bone，contrast，cool，copper，flag，gray，hot，hsv，jet，lines，prism，spring，summer，winter，orient，pagesetupdlg，print，printdlg，printopt，saveas，allchild，copyobj，findall，findobj，gcbo，gco，get，ishandle，rotate，set，getappdata，isappdata，rmappdata，setappdata，axes，figure，image，light，line，patch，rectangle，surface，text，uicontextmenu，capture，clc，clf，close，closereg，gcf，newplot，refresh，saveas，axis，cla，gca，reset，rotate3d，selectmoveresize，ginput，zoom，dragrect，drawnow，rbbox，coneplot，contourslice，curl，divergence，interstreamspeed，isocolors，isosurface，streamparticles，streamribbon，streamslice，sreamtube，volumebounds
gui	图形用户界面	dialog，errordlg，helpdlg，inputdlg，listdlg，msgbox，pagedlg，pintfdlg，questdlg，uigetfile，uiputfile，uisetcolor，uisetfont，warndlg，inspect，movequi，openfig，quidata，quide，quihandles，clf，close，figure，gcf，axes，axis，caxis，cla，gca，hold，subplot，axes，figure，image，line，patch，surface，text，uicontextmenu，uicontrol，uimenu，delete，drawhow，findobj，gco，get，newplot，reset，set，capture，orient，print，printopt，getframe，movie，moviein，dragrect，findfigs，gcbf，gcho，ginput，graymon，ishold，rbbox，rotate，selectmoveresize，terminal，textwrap，uigetfile，uiputfile，uiresume，uiwait，waitbar，waitforbuttonpress，whitebg，zoom，dialog，figflag，layout，uiguide，prtps，prtwin

参 考 文 献

1 陈永春．MATLAB M 语言高级编程．北京：清华大学出版社，2004

2 朱横君．MATLAB 语言及实践教程．北京：清华大学出版社；北京交通大学出版社，2005

3 刘宏友等．MATLAB6 基础及应用．重庆：重庆大学出版社，2002

4 导向科技．MATLAB 6.0 程序设计与实例应用，北京：中国铁道出版社，2001

5 王沫然．MATLAB 与科学计算．第 2 版．北京：电子工业出版社，2004

6 苏金明等．MATLAB 与外部程序接口．北京:电子工业出版社，2004

7 苏金明等．MATLAB 7.0 实用指南．北京：电子工业出版社，2004

8 郝文化．MATLAB 图形图像处理应用教程．北京：中国水利水电出版社，2004

9 施阳等．MATLAB 语言工具箱．西安：西北工业大学出版社，1998

10 邹鲲等．MATLAB6.x 信号处理．北京：清华大学出版社，2002

11 楼顺天等．基于 MATLAB 的系统分析与设计：神经网络．西安：西北电子科技大学出版社，1998

12 夸克工作室．有限元分析基础篇-ANSYS 与 MATLAB．北京：清华大学出版社，2002

13 施阳等．MATLAB 语言精要及动态仿真工具 SIMULINK．西安：西北工业大学出版社，1997

14 沈辉．精通 SIMULNK 系统仿真与控制．北京：北京大学出版社，2003

15 张立明．人工神经网络的模型及其应用．上海：复旦大学出版社，1994

16 Lindfield G，Penny J．Numerical Methods Using MATLAB．2nd ed．NJ：Prentice Hall，2000